Statistical Classification Based Modelling and Estimation of Analog Circuits Failure Probability

Muhammad Shirjeel Shehzad

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science (Electrical & Computer Engineering)

Concordia University

Montréal, Québec, Canada

December 2016

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: Muhammad Shirjeel Shehzad

Entitled: "Statistical Classication Based Modelling and Estimation of Analog Circuits Failure Probability"

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. Rabin Raut

_____ Examiner
Dr. Rajagoplan Jayakumar

_____ Examiner
Dr. Pouya Valizadeh

_____ Supervisor
Dr. Sofiène Tahar

Approved by _____
Dr. William E. Lynch, Chair
Department of Electrical and Computer Engineering

_____
Dr. Amir Asif, Dean
Faculty of Electrical and Computer Engineering

Date _____

Abstract

Statistical Classification Based Modelling and Estimation of Analog Circuits Failure Probability

Muhammad Shirjeel Shehzad

At nanoscales, variations in transistor parameters cause variations and unpredictability in the circuit output, and may ultimately cause a violation of the desired specifications, leading to circuit failure. The parametric variations in transistors occur due to limitations in the manufacturing process and are commonly known as process variations. Circuit simulation is a Computer-Aided Design (CAD) technique for verifying the behavior of analog circuits but exhibits incompleteness under the effects of process variations. Hence, statistical circuit simulation is showing increasing importance for circuit design to address this incompleteness problem. However, existing statistical circuit simulation approaches either fail to analyze the rare failure events accurately and efficiently or are impractical to use. Moreover, none of the existing approaches is able to successfully analyze analog circuits in the presence of multiple performance specifications in timely and accurate manner. Therefore, we propose a new statistical circuit simulation based methodology for modelling and estimation of failure probability of analog circuits in the presence of multiple performance metrics. Our methodology is based on an iterative way of estimating failure probability, employing a statistical classifier to reduce the number of simulations while still maintaining high estimation accuracy. Furthermore, a more practical classifier model is proposed for analog circuit failure probability estimation.

Our methodology estimates an accurate failure probability even when the failures

resulting from each performance metric occur simultaneously. The proposed methodology can deliver many orders of speedup compared to traditional Monte Carlo methods. Moreover, experimental results show that the methodology generates accurate results for problems with multiple specifications, while other approaches fail totally.

To My Parents

Acknowledgments

Firstly, I would like to thank Dr. Sofiène Tahar for his help, guidance and encouragement throughout my Master's degree. I am very grateful to him for his support in my research, sound advice, insightful criticisms and prompt feedback. Other than research, I have learned many practical and professional aspects from him.

I would like to thank all members of my examining committee, Dr. Rajagopalan Jayakumar, Dr. Pouya Valizadeh and Dr. Rabin Raut.

It was not possible for me to complete my research without the support and help of Ons Lahiouel. It was her continuous motivation and support that helped me achieve my goals during the hard times. My sincere thanks to all my friends in the Hardware Verification Group for their support and motivation and for the good times spent together inside and outside the lab. Moreover, I would like to thank Ons Lahiouel, Mbarka Soualhia and Mahmoud Masadeh for taking the time to go over my thesis.

I am deeply thankful to my parents and my sister for their never ending love, inspiration and encouragement. It was their belief in me that drives me through any situation and makes me accomplish all my goals. Also, my sincere thanks and appreciation to my best friends, Shahzad Khan and Qazi Sibghat-Ul-Haq. It was a long and challenging journey and they made it easy and enjoyable for me.

Contents

List of Figures

List of Tables

List of Acronyms

BL	Bit Line
CAD	Computer Aided Design
CDF	Cumulative Distribution Function
DC	Direct Current
EDA	Electronic Design Automation
GBW	Gain BandWidth
GPD	Generalized Pareto Distribution
GRBF	Gaussian Radial Basis Function
HNM	Hold Noise Margin
IC	Integrated Circuit
K-SVM	Kernel Support Vector Machines
L-SVM	Linear Support Vector Machines
LER	Line Edge Roughness
LFR	Likely Failure Region
LHS	Latin Hypercube Sampling
MC	Monte Carlo
MLE	Maximum Likelihood Estimation
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
PDF	Probability Distribution Function
PPM	Parts Per Million
PVT	Process, Voltage and Temperature
PWM	Probability-Weighted Moment
RCO	Random Crystal Orientation
RDF	Random Dopant Fluctuation
RNM	Read Noise Margin
RSB	Recursive Statistical Blockade
SB	Statistical Blockade

SNM	Static Noise Margin
SPICE	Simulation Program with Intergraded Circuit Emphasis
SRAM	Static Random Access Memory
SVM	Support Vector Machines
TSMC	Taiwan Semiconductor Manufacturing Company
TTM	Time-To-Market
TTP	Time-To-Product
ULFR	Unlikely Failure Region
VLSI	Very Large Scale Integrated
VTC	Voltage Transfer Curve
WL	Word Line
WNM	Write Noise Margin
QMC	Quasi Monte Carlo

Chapter 1

Introduction

1.1 Motivation

The need to improve quality of life has been the driving force for innovation in the semiconductor industry. As a result of these innovations a processing unit, which used to be the size of a room, is now the size of a finger nail because of down to nanometer scaling in transistor size. Smaller transistors mean a larger number of transistors can be fabricated on a single wafer of silicon. Over the past five decades, the number of transistors on a chip has increased exponentially in accordance with Moore's law [1]. This has resulted in the design of complex Very Large Scale Integrated (VLSI) circuits. However, at such small sizes, even small variations due to the random nature of the manufacturing process can cause large relative variations in the behavior of a circuit. Based on the source of variation, such variations can be broadly classified into two categories: (1) *systematic variation*, and (2) *random variation*. Systematic variations represent the deterministic part of these variations; e.g., proximity-based lithography effects, etc. [2]. Systematic variations are typically pattern dependent and can potentially be completely explained by using more accurate models of the process. Random variations make up the unexplained part of the manufacturing variations, and show stochastic/random behavior, e.g., gate oxide thickness (t_{ox}) variations, Random Crystal Orientation (RCO) and Random Dopant

Fluctuation (RDF) [3].

Random variations in the manufacturing process are more commonly known as *process variations*. Process variations cause unpredictability and variations in the circuit output, and may ultimately cause violation of the desired specifications, leading to circuit failure. In fact, failures in critical circuits may lead to failures in the entire chip. Moreover, in the complex VLSI designs, the designer must deal with hundreds of process parameters for custom circuits and millions for chip-level designs. Therefore, verifying the circuit behavior in the presence of process variations has become an area of major concern.

A typical system-on-chip [4] consists of different analog, digital and mixed signal circuitry. A mixed signal circuit combines analog and digital behavior on a single integrated circuit [5]. In this thesis, we focus on verifying the behavior of analog circuits and the analog part of mixed signal circuits under the influence of process variations because of the complex and knowledge intensive nature of these circuits [6].

Many of the Electronic Design Automation (EDA) tools for modeling and simulating analog circuit behavior, are unable to accurately model and predict the large impact of process-induced variations on the circuit behavior. Recently, formal methods [7] have been investigated for verifying analog circuits under the influence of process variations but have found limited practical use because of the fact that the state space of analog circuit models is infinite [8]. Behavioral models are also used for verifying analog circuits in which the process variations are considered as initial conditions. However, to make the problem manageable, different levels of approximation are considered in the behavioral model [9].

When other attempts for verifying analog circuits started failing, *statistical analysis* approach was adopted [9]. Statistical analysis is the science of collecting and exploring large amounts of data to discover hidden patterns. In statistical analysis of circuits, samples are taken from distributions of process parameters. Each sample is a set of values for the parameters of a circuit, which can occur due to process variations.

These samples are then simulated using circuit simulators [10], giving samples of the circuit's output. The probability that the circuit does not meet performance specifications is estimated by analysing the trend in output samples. Statistical circuit simulation is displaying a considerable and increasing importance for circuit design under process variations [9]. One standard approach of a statistical circuit simulation is the repeated drawing of random samples from distributions of process parameters and simulating the samples (Monte Carlo method [11]). To obtain accurate results, a large number of samples should be simulated. Circuit simulators employ numerical evaluation of mathematical models of the circuits. Accurate analog circuit models are usually very complex as they capture the effect of nonlinearities in semiconductor devices, technology scaling, and Process, Voltage and Temperature (PVT) variations [12].

A lot of efforts have been exerted to reduce the runtime of a single simulation [13, 14, 15]. However, for the case of a very low failure probability like 10^{-6}, millions of samples should be simulated to capture one single failure. One may ask why not simply ignore such a small failure. Consider the case of a 1-Megabit (Mb) memory array which is an example of mixed signal designs. The memory array has 1 million identical instances of a memory cell. Designed to be identical, but due to the stochastic behavior of manufacturing process, they usually differ. If a memory cell failure causes the overall memory chip failure and a yield rate of 99% is required, i.e., no more than one memory chip per 100 should fail. This means that on average, not more than one per million cells should fail. This translates to a required yield of 99.999999, or a maximum failure rate of 0.01 Parts Per Million (ppm) for the single cell. In such scenarios, even the very small failure probability of the circuit has to be estimated, to determine its effects on a chip level design. Factors like Time-To-Market (TTM) and Time-To-Profit (TTP) [16] and customer satisfaction require that the analog circuit verification must be performed accurately and efficiently.

Over time, more advanced statistical approaches have been proposed for efficiently estimating the probability of rare failure events. Existing approaches either fail to

analyze the rare failure events accurately and efficiently or are practically infeasible. Moreover, none of the existing approaches is able to successfully analyze circuits in the presence of multiple specifications, accurately and efficiently.

The general motivation of this thesis is to present a statistical circuit simulation based methodology for modelling and estimating failure probability of analog circuits with multiple performance specifications. In our methodology, we propose several enhancements to the existing approaches either by developing new statistical tools or by employing existing advanced statistical tools. Our methodology is more accurate and practical to use than any of the existing ones. We also provide a complete formulation of the failure probability estimation in the presence of multiple performance metrics.

1.2 Related Work

Process variation has made circuit reliability an area of growing concern in modern times. Statistical based circuit simulation approaches have been adopted to estimate the likelihood of a circuit not meeting its desired specifications. One golden standard approach to estimate failure in probabilistic circuit performance is Monte Carlo (MC) [11]. Apart from MC, other fast statistical approaches have been proposed in the past decade. In this section, we provide an overview of these approaches and highlight their strengths and weaknesses. In particular, we can categorize statistical approaches related to analog circuit failure probability estimations into four main categories: Monte Carlo and its variants, Moment matching, Importance sampling and Statistical classification.

1.2.1 Monte Carlo and its Variants

Over the years, Monte Carlo (MC) has become a standard technique for statistical simulation of circuits and for yield estimation during the design phase [17, 18, 9]. MC methods in their simplest form are referred to as naive, crude or traditional

MC. Using naive MC, random samples are drawn repeatedly from the distributions of process variation parameters and circuit performance is evaluated for the samples using SPICE simulations [19]. The failure probability is then estimated using the following formula:

$$\text{Failure Probability} = \frac{\text{Number of samples not meeting the desired specification}}{\text{Total number of samples drawn}} \quad (1)$$

A large number of samples/simulations are required for accurate failure probability estimation using naive MC hence it is highly time consuming. Moreover, millions of samples need to be simulated to capture a single failure when failures are rare events, making its runtime prohibitive. To relieve the problem, Latin Hypercube Sampling (LHS) [20] and Quasi Monte Carlo (QMC) [21] have been proposed. LHS aims to spread the sample points more evenly across all possible values by dividing the distribution to sub-intervals. QMC generate quasi-random numbers rather than purely-random samplings, which can save a large number of samples. However, the performance of QMC can degrade for high dimensional problems [22], where each process parameter is representing a dimension. QMC and LHS may save samples, but the samples requirement for rare failure events is still comparatively large.

1.2.2 Moment Matching

Moment matching is a method of estimating population parameters (moments) using samples of the population. In moment matching based approaches [23, 24] for analog circuit verification, a small number of samples are simulated using SPICE. Simulation results along with process parameters samples are used for evaluating moments of a performance metric. Then the Probability Density Function (PDF) of the performance metric is approximated to an analytical expression using a moment matrix. This moment matrix is evaluated using conventional methods of moment matching. For high dimensional problems, the condition number for the moment matrix becomes too large, making it numerically unstable [25]. MAXNET [26] overcomes the issue of dimensionality by only considering the behavior of the performance metric as its sole

input. However, all moment matching approaches model overall PDF only without surgically looking into tail region. Tail is of great importance as it contains information special to rare events [27]. Therefore, moment matching based approaches are only used to analyze overall circuit behavior rather than estimating rare failure events.

1.2.3 Importance Sampling

Importance sampling [28] based approaches were developed to overcome the problem of rare failure event estimation [29]. In importance sampling based approaches for analog circuit failure probability estimation, the distributions of process variation parameters are shifted to the failure region to help MC methods draw more samples from rare failure events. These samples are simulated using SPICE, reweighted and then are used to calculate the rare failure events probability. Minimum L_2-norm [30] is used for shifting distribution in approaches proposed by [31, 32, 33] while particles filters are adopted in [34] to help MC methods draw samples from failure regions. All of these approaches assume a single failure region in the parameter space. In reality, there may exist multiple failure regions. Moreover, the reweighting process becomes regenerate and unbounded with increased dimensions [35, 36].

The approach proposed in [37] overcomes limitations of the existing importance sampling approaches by first clustering the parameters space into hyperspaces and then drawing samples from these clusters. However, results obtained from every run are different and multiple runs are required to obtain an accurate estimate. Another approach is proposed in [38] to overcome the problem of multiple failure regions. First the failure regions are explored and then, by performing importance sampling on these regions, failure probability is estimated. However, this approach failed in high dimensions since it depends on a surrogate model [39] instead of SPICE simulations, for finding failure regions.

1.2.4 Statistical Classification Based Methods

Statistical classification is a way of predicting the category/class of input data on the basis of training data, containing observations whose category membership is known [40]. The function that predicts the membership of input data is called a classifier. For the case of analog circuit failure probability estimation, a classifier is employed to categorize a sample of process parameters as likely-to-fail or unlikely-to-fail without performing SPICE simulation. Likely-to-fail samples are those samples of process parameters, which are likely to cause circuit failure. While unlikely-to-fail samples are unlikely to cause circuit failure. Unlikely-to-fail samples are discarded while other samples are simulated using SPICE. The results of these simulation represent tail regions of the distribution of the performance metric and are modeled using Generalized Pareto Distribution (GPD). GPD is a type of probabilistic distribution used to model the tail region of another distribution [41].

A classification based approach was first proposed by Statistical Blockade (SB) [42], making use of a Linear Support Vector Machine (L-SVM) classifier. Recursive Statistical Blockade (RSB) [27] further enhanced the SB method, by an iterative estimation of failure regions using the L-SVM classifier. However, neither a single L-SVM is sufficient to deal with non linear boundaries of failure regions nor it can effectively deal with multiple failure regions [33]. In [43] a non-linear classification approach called REscope was adopted to overcome the issues of SB. Parameter pruning based on initial sample selection was also applied to only focus on critical process parameters while ignoring others. However, parameters pruned may be the real sensitive parameters in failure regions. Furthermore, REscope requires quite a large number of simulations for estimating the probability of extremely rare failure events. Therefore, in [44], an approach called Smartera proposed the use of a non-linear classifier in an iterative way based on the RSB method to estimate the probability of extremely rare failure events.

Both REscope and Smartera employed a Gaussian Radial Basis Function (GRBF) based Kernel Support Vector Machines (K-SVM) as the non-linear classifier. K-SVM

requires choosing a value for the kernel scale parameter [45]. Choosing the correct value of the kernel scale parameter requires many iterations of training and testing. Even then, the validity of the value chosen for kernel scale parameter cannot be completely verified until tested against a large data set. Hence, limiting the K-SVM ability for analog circuit verification from a practical point of view. Moreover, none of the statistical classification based approaches was able to verify the validity of their methodology/framework in presence of multiple performances metrics. Also the problem of overlapping of failure events resulting from different performance metrics remains completely unaddressed.

1.3 Proposed Methodology

In the related work section, we briefly discussed statistical approaches for analog circuit verification. As indicated, all of the approaches have some kind of limitations. The main objective of this thesis is to develop a general methodology/framework overcoming the limitations of these approaches. In particular, we propose to develop a methodology characterized by the ability to:

- reduce time required for estimation rare failure event probability without compromising on accuracy.
- handle a large number of process variation parameters.
- provide complete failure region coverage in the process parameters space.
- deal with multiple performance metrics.
- deal with the overlapping of failure events resulting from each performance metric.
- become widely applicable, i.e., can be adopted by designers in industries.

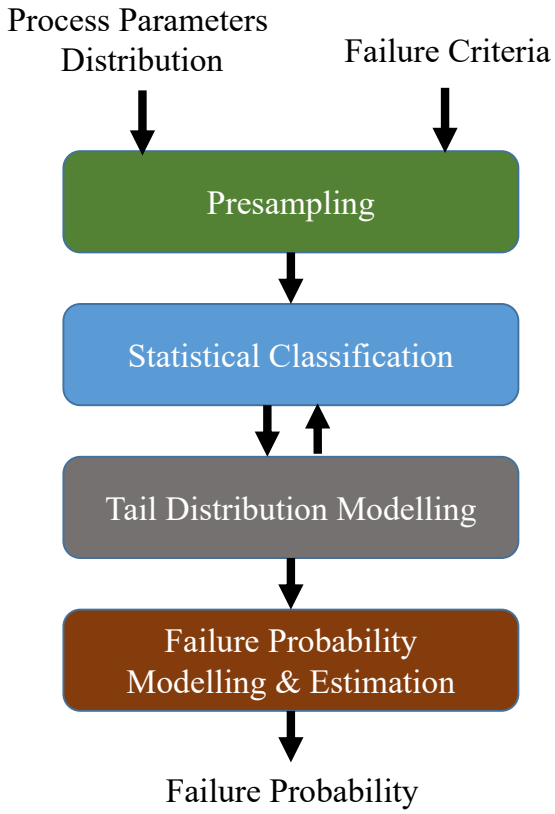

Figure 1.1: Methodology to Estimate Failure Probability of Analog Circuits.

Figure 1.1 shows a simplified block diagram of our methodology. Our methodology falls into the category of statistical classification based methods. The methodology consists of four processes: (1) presampling; (2) statistical classification; (3) tail distribution modelling; and (4) failure probability modelling and estimation. The inputs to our methodology are distributions of the process variation parameters and circuit specifications defined in term of failure criteria for performance metrics. In the first process, we use LHS and moment matching methods to model the overall performance metric distributions. In the second process, samples that are likely to cause circuit failure are determined using a statistical classifier. In the third process, these likely-to-fail samples are simulated using SPICE and results are modelled as

tail region of the overall distribution by Generalized Pareto Distribution (GPD) fitting. The process of classification and GPD fitting is repeated iteratively to get a better model of tail distribution while reducing sample counts. Finally in the last process, the methodology estimates the rare failure probability of the circuit using the overall distribution and GPD. The output of the methodology is the estimated failure probability.

The proposed methodology estimates the circuit failure probability by analyzing the circuit behavior at the transistor level design. A large analog design with many transistors is not verified as a whole, but is decomposed into sub-blocks. Each sub-block is then further decomposed down to the cell level. A cell is an analog circuit having a certain basic function and the failure probability of the cell is estimated by the proposed methodology.

We also illustrate the application of our proposed methodology on various analog circuits to prove its effectiveness. The circuits used for this purpose, namely, are a ring oscillator, an operational amplifier (opamp) and a Static Random Access Memory (SRAM) cell. We use the opamp and SRAM cell circuits to verify the validity of our methodology to estimate the failure probability of analog circuit in the presence of multiple performance specifications. The ring oscillator circuit is used to verify that our methodology is suitable for high dimensional problems. Our methodology estimates the failure probabilities of all three circuits based on their specifications. We provide an in-depth analysis of obtained results and justify the use of various techniques proposed in our methodology. We also compare the obtained results with other methods, namely, the naive MC method, REscope and Smartera.

1.4 Thesis Contributions

In this thesis, a comprehensive failure probability modeling and estimation methodology for the analog circuits is presented. The contributions of the thesis can be summarized as follows:

- We reduced the number of samples required for estimating the failure probability of analog circuits.
- We developed a more practical statistical classifier for analog circuits dataset, which proved to be more efficient and accurate than previously used classifiers for analog circuit failure probability estimation.
- We derived the mathematical formulas to calculate the failure probability in the presence of multiple performance specification and overlapping failure events and developed an algorithm to estimate the probability of overlapping failure events.
- We conducted experiments on three analog circuits to estimate their failure probability in the presence of process variation and multiple performance specification and compared results with other recently published work.

1.5 Thesis Organization

The rest of the thesis is organized as follows: In Chapter 2, we present a brief overview of the concepts and techniques used in this thesis. Then, in Chapter 3, we explain the proposed methodology with an overall flow and detailed description of the statistical classification process developed and mathematical formulation for multiple performance specification. The chapter also describes the algorithm developed for estimating the probability of overlapping failure events due to multiple performance specifications. In Chapter 4, we use the proposed methodology to estimate the failure probability of three test circuits and compare the obtained results with other approaches. Finally, Chapter 5 provides conclusions of this thesis and directions for future research.

Chapter 2

Preliminaries

In this chapter, we start by giving some basic concepts in probability theory. We briefly describe the notion of Support Vector Machines (SVM) for statistical classification. Subsequently, we provide an overview of rare event modelling on which our methodology is based. Finally, we present an overview of Latin Hypercube Sampling (LHS), SPICE simulator and k-means clustering which are used in our methodology. The intent of this chapter is to introduce the basic theories and concepts that we use in the rest of this thesis.

2.1 Basic Concepts in Probability Theory

The basic definitions and concepts in probability are briefly reviewed in this section. These concepts are essential for the understanding of statistical failure probability estimation of analog circuits.

2.1.1 Random Variable and Random Process

A random or stochastic variable is a variable (like other mathematical variables) that we cannot say for sure which value it will take on. However, the value that a random variable can take on can be associated with a probability of the value. There are two kinds of random variables: discrete random variables and continuous random

variables. A discrete random variable can take on values from a finite or countably infinite set of numbers, e.g., result of a coin toss. A continuous random variable can take on values from an interval of real numbers, e.g., any value within the interval [0,1] can be assumed by continuous random variable. Normal or Gaussian random variables are the most commonly encountered continuous random variable in both manmade and natural phenomena. Process variation parameters are also continuous random variable.

A random or stochastic process is a function that produces a random variable as an output. It is the probabilistic counterpart of a deterministic process in which output values can be known for sure given the input and initial conditions.

2.1.2 Distribution Function

A probability distribution is a table or an equation that links each value assumed by the random variable in an experimental setting with its probability of occurrence. A probability distribution can be specified in a number of different ways, of which most common are the Probability Distribution Function (PDF) and the Cumulative Distribution Function (CDF). The choice of the distribution function is based on mathematical convenience. Let X be a random variable that can take any value x in the interval $(-\infty, \infty)$ with probability $Pb(x)$, then the CDF is given by:

$$F(x) = Pb(X \leq x) = \int_{-\infty}^{x} f(t)dt \tag{2}$$

CDF is a positive and monotonically increasing bounded function and $F(\infty) = 1$. The PDF of X is given by the following relation:

$$f(x) = \frac{dF(x)}{dx} \tag{3}$$

The PDF of a gaussian random variable is given by the following equation:

$$f(x) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4}$$

where μ is the mean and σ is the standard deviation of the distribution.

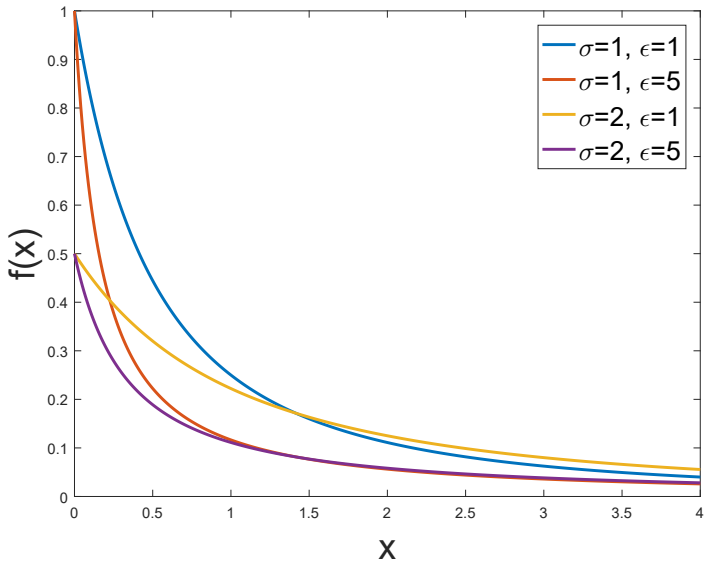

Figure 2.1: Generalized Pareto Distribution for Different Values of σ & ϵ, $\mu = 0$.

2.1.3 Generalized Pareto Distribution

The Generalized Pareto Distribution (GPD) is a family of continuous probability distributions often used to model the tails of another distribution. Tail refers to the part of distribution which is quite far away from the mean. GPD allows a continuous range of possible shapes that includes both the exponential and Pareto distributions as special cases. The CDF of GPD is given by the following equation:

$$F(x) = \begin{cases} 1 - (1 - \frac{\epsilon(x-\mu)}{\sigma})^{\frac{1}{\epsilon}} & for\ \epsilon \neq 0 \\ 1 - exp(1 - \frac{\epsilon(x-\mu)}{\sigma}) & for\ \epsilon = 0 \end{cases} \tag{5}$$

where μ is the starting point, σ is the scale parameter, and ϵ is the shape paramter. Figure 2.1 shows the CDF of GPD for different values of σ and ϵ.

2.2 Support Vector Machines Classifier

A Support Vector Machine (SVM) is a classifier (Section 1.2.4) defined by a hyperplane separating different categories of data points in the given dataset. A hyperplane

is a subspace of one dimension less than its ambient space. We provide an introductory discussion of the basic ideas behind SVMs in this section. SVMs are among the best supervised learning algorithms. By supervised, we mean that SVMs first have to be trained for future predictions.

To explain how the SVM works, we consider a 2-dimensional dataset having two classes/categories of points, as shown in Figure 2.2. The figure also shows multiple lines, separating the two categories. These lines are actually hyperplanes which can be used to categorize new data points. But which is the best among all hyperplanes?

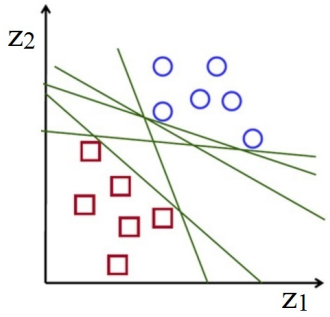

Figure 2.2: Multiple Lines Separating two Classes of 2D Dataset.

A hyperplane is bad if it passes too close to the data points because it will be noise sensitive. Thus the best solution is to find a hyperplane which is as far as possible from data points while still providing a separation between different categories of the data points. The two dotted hyperplanes shown in Figure 2.3 provide the maximum separation between categories. The distance between these hyperplanes is called

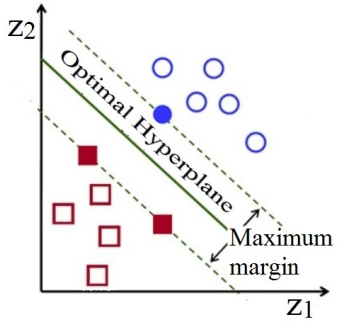

Figure 2.3: Optimal Hyperplane Separating two Classes of 2D Dataset.

margin. An optimal hyperplane is the hyperplane that lies halfway between the margin. The operation of the SVM algorithm is to search for this optimal hyperplane, that maximizes the margin of the training data.

2.2.1 Linear SVM

The dataset shown in Figures 2.2 and 2.3 can be separated using a linear hyperplane. An SVM classifier that separates different categories of a dataset using linear hyperplane is called a Linear SVM (L-SVM) classifier. To understand how the linear hyperplane is determined, suppose that we are given a training dataset of n points of the form:

$$(\vec{z_1}, y_1), ..., (\vec{z_n}, y_n)$$

Here, y_i represents the class of the point $\vec{z_i}$, with value either 1 or -1. Each $\vec{z_i}$ is a p-dimensional vector. In this example, a binary classification problem is considered because the classification process adopted in the methodology proposed in this thesis is also binary.

The formal notion to represent a hyperplane is given by the following relation [45]:

$$\vec{w}.\vec{z} - b = 0 \tag{6}$$

where $\vec{w}$ represents the normal vector to a hyperplane. The parameter $\frac{b}{||w||}$ gives the distance of the hyperplane from the origin in the direction of $\vec{w}$.

If the training dataset is linearly separable (different classes can be separated by a linear hyperplane), the two hyperplanes shown in Figure 2.3, which provide the maximum separation between different categories of dataset are given by the following equations [45]:

$$\vec{w}.\vec{z} - b = 1$$

and

$$\vec{w}.\vec{z} - b = -1$$

The distance between these two hypeplanes can be evaluated geometrically, which equals to $\frac{2}{||w||}$. Then to maximize distance between the two classes, $||w||$ has to

be minimized. For mathematical convenience, we state the problem as minimizing $\frac{1}{2}||w||^2$. Moreover, while minimizing, to prevent data points from falling into the margin, we add a constraint to each data point $\vec{z_i}$ either

$$\vec{w}.\vec{z_i} - b \geq 1, \text{ if } y_i = 1$$

or

$$\vec{w}.\vec{z_i} - b \leq -1, \text{ if } y_i = -1$$

Above mentioned constraint can be rewritten as:

$$y_i(\vec{w}.\vec{z_i} - b) \geq 1 \text{ , for all } 1 \leq i \leq n. \tag{7}$$

Putting together the problem of minimizing $\frac{1}{2}||w||^2$ and the constraint given in Equation 7, we have the following optimization problem:

$$\begin{aligned} &\text{Minimize } \frac{1}{2}||w||^2 \\ &\text{subject to } y_i(\vec{w}.\vec{z_i} - b) \geq 1, \text{ for } i = 1, ...n \end{aligned} \tag{8}$$

This is a problem of Lagrangian optimization that can be solved using Lagrange multipliers. It is given by [45]:

$$\min L(w) = \frac{1}{2}||w||^2 - \sum_i \alpha_i \Big[y_i(\vec{w}.\vec{z_i} + b) - 1 \Big] \tag{9}$$

An important consequence of the geometric description is that the max-margin hyperplane is determined only by those data points which are nearest to it. These data points are called support vectors. The value of α_i in Equation 9 for the datapoint z_i is zero unless z_i is a support vector.

The $\vec{w}$ and b that solve the problem given by Equation 9, determine our linear classifier. Given the solution of Equation 9, the decision function of the classifier can be written as:

$$G(z) = sign(\vec{z}.\vec{w} + b) \tag{10}$$

2.2.2 Kernel SVM

Evaluating the derivative of Equation 9 with respect to $\vec{w}$ and b and setting them equal to zero, to determine *extrema*, we have:

$$\vec{w} = \sum_i \alpha_i y_i \vec{z_i} \tag{11}$$

$$\sum_i \alpha_i y_i = 0 \tag{12}$$

Substituting values from Equations 11 and 12 in Equation 9, we get:

$$\min\ L(w) = \sum_i \alpha_i\ - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j (\vec{z_i}.\vec{z_j}) \tag{13}$$

and substituting values from Equations 11 and 12 in Equation 10, we get:

$$G(z) = sign(\sum_i \alpha_i y_i (\vec{z_i}.\vec{z}) + b) \tag{14}$$

From Equations 13 and 14, we determine that the optimization and solution for classification, both depends upon the *dot product*. If the dataset is not linearly separable in the current feature space, then transforming the feature space into high dimensional space, can make linear separation possible. If the transformed space is given by $\Phi(\vec{z})$, then according to Equations 13 and 14, only $\Phi(\vec{z_i}).\Phi(\vec{z_j})$ and $\Phi(\vec{z_i}).\Phi(\vec{z})$ are required for the training and prediction in the transformed space. Furthermore, if we have a function $k(\vec{z_i}, \vec{z_j}) = \Phi(\vec{z_i}).\Phi(\vec{z_j})$, the classification can be done in the transformed feature space without actual transformation. This function is called the *kernel function*. The kernel function allows SVM to perform non-linear classification. An SVM classifier based on the kernal function is called Kernel SVM (K-SVM). Following two are the most common kernel functions [46]:

- Polynomial: $k(\vec{z_i}, \vec{z_j}) = (\vec{z_i}.\vec{z_j} + 1)^d$
- Gaussian Radial Basis Function: $k(\vec{z_i}, \vec{z_j}) = exp(-\gamma||\vec{z_i} - \vec{z_j}||^2)$, for $\gamma > 0$.

2.3 Rare Event Modelling

In Section 1.3, it was stated that the overall distribution of performance metric and its tail region is modelled to estimate rare failure events in our methodology. In this section, we explain how a rare event is modelled and estimated in our methodology Consider an analog circuit with a performance metric Y. Because of the variation in manufacturing process, parameters of the test circuit are considered random variables with joint probability distribution S. This in turns makes Y also a random variable since its value depends on parameters values. If our specification requires that any value of Y which is greater than the failure criteria t_f causes circuit failure, then the failure probability Pb_f of the analog circuit is given by:

$$Pb_f = Pb(Y > t_f) = 1 - F(t_f) \tag{15}$$

where $F(t)$ is the CDF of Y. If t_f represents some rare event in the distribution of Y, then Pb_f represents a rare failure probability. Suppose Y belongs to a Gaussian distribution with PDF $f(y)$ given by Equation 4. Suppose t is a threshold that separates the tail from the body of the PDF $f(y)$ and lies between the mean and t_f as shown in Figure 2.4(a), the probability of event $Y > t$ is given by:

$$Pb(Y > t) = 1 - F(t) \tag{16}$$

Let z be an excess over t. Using the concepts of conditional probability [47], the conditional CDF is given by:

$$F_t'(z) = Pb(Y - t \leq z | Y > t) = \frac{F(z+t) - F(t)}{1 - F(t)} \tag{17}$$

and the overall CDF as:

$$F(z+t) = (1 - F(t))(F_t'(z)) + F(t) \tag{18}$$

Now if z represents $t_f - t$ as shown in Figure 2.4(b), then we have:

$$F(t_f) = (1 - F(t))(F_t'(t_f - t)) + F(t) \tag{19}$$

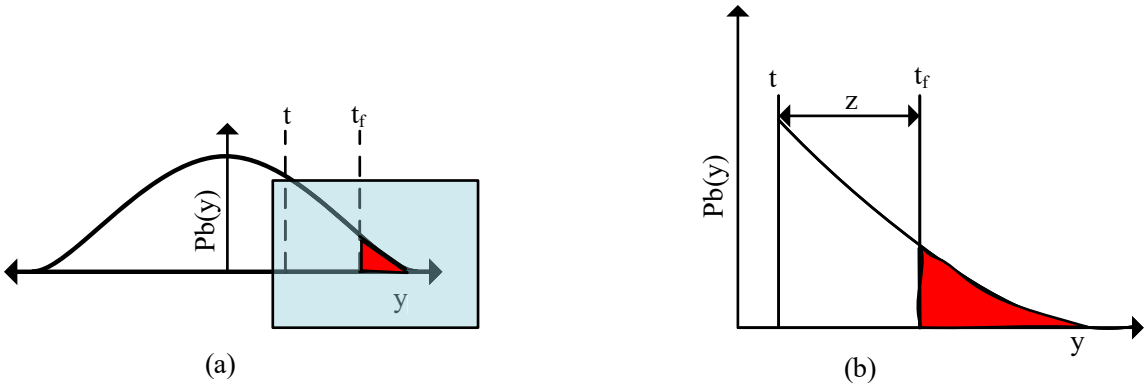

Figure 2.4: Rare Event Modelling using Tail Distribution.

substituting the value of $F(t_f)$ given by Equation 19 in Equation 15 and rearranging, we have:

$$Pb_f = (1 - F(t))(1 - F'_t(t_f - t)) = Pb(t)Pb(Y > t_f | Y > t) \tag{20}$$

$F'_t(t_f - t)$ represents the probability of exceedence and can be modelled by GPD [41] with $\mu = t$. If $F_t(y)$ represents the CDF of GPD, we have [9]:

$$Pb_f = Pb(t)Pb(Y > t_f | Y > t) = (1 - F(t))(1 - F_t(t_f)) \tag{21}$$

Using Equation 21, we can efficiently estimate the failure probability. $F(t)$ can be accurately estimated using few hundred simulation, using the methods of moment matching (Section 1.2.2). Once estimated, we can model $F_t(y)$ only by simulating samples in the region $Y > t$, blocking all other samples. But we do not know which samples of the parameter space when simulated will generate values of the performance metric Y greater than t. Hence one can employ a statistical classifier to block all samples from being simulated that are unlikely to generate values of Y greater than t. By doing so, an accurate estimation can be made with a small number of simulations. In our methodology t is considered as a relaxed failure criteria while t_f is an actual failure criteria, i.e., the failure criteria given by the designer. Therefore, the classifier is trained on the basis of a relaxed failure.

While deriving above relation, we assume that the extreme values of interest lie only in the upper tail of the distribution. This is without loss of generality because any lower tail can be converted to the upper tail by replacing $y = -y$.

2.4 Latin Hypercube Sampling

Latin Hypercube Sampling (LHS) first introduced in 1979, is a sampling method for generating a near-random sample from a multidimensional distribution. In the context of analog circuit verification, LHS is employed for the purpose of variance reduction in the distribution of process variation parameters.

In a 2-dimensional space, a square grid containing sample positions is a Latin square if (and only if) there is only one sample in each row and each column. A Latin hypercube is the generalisation of the concept of Latin square to an arbitrary number of dimensions. When sampling from N variate distribution, LHS partitions the distribution into M intervals of equal probability, and selects one sample from each interval. This forces the number of divisions, M, to be equal for each variable. Moreover, samples for each input are shuffled so that there exists no correlation between the inputs.

2.5 SPICE

Simulation Program with Integrated Circuit Emphasis (SPICE) [19] is powerful general purpose analog circuit simulator, which is used to predict the circuit behavior and to verify circuit design. SPICE can simulate components ranging from the basic passive elements such as resistors, capacitors and inductors to more sophisticated semiconductor devices such as MOSFETs. Using these intrinsic components as the basic building blocks, very large and complex circuits can be simulated in SPICE. SPICE can perform several types of circuit analysis. Some of the important ones are:

- DC analysis

- Transient analysis
- AC analysis
- Noise analysis
- Sensitivity analysis
- Distortion analysis
- Fourier analysis
- Monte Carlo Analysis

Moreover, using SPICE, the analysis can also be performed for different temperatures. SPICE employs complex transistors and other circuit elements models to predict accurate behaviors.

2.6 k-means Clustering

Data clustering is a type of unsupervised learning, that divides a set of objects into groups or cluster in a way that objects of the same group exhibit a certain measure of similarity [45]. One of the most used and popular clustering algorithm is k-means [48]. k-means classifies input objects into predefined number of clusters. Figure 2.5 shows the simplified flowchart of k-means clustering algorithm. The inputs to the algorithm are n objects and a value k that represents the number of clusters. Initially, the algorithm randomly chooses k cluster centroids. For all objects, the distance from each of the centroids is evaluated. The input objects are then assigned to the group associated with the nearest centroid. Based on the members of a group, a new centroid is evaluated for each group. For all objects, the distance from each of the new centroid is evaluated. Based on the minimum distance from new centroids, the membership of the object to a group is then updated. This process is repeated iteratively until no object is reassigned to a new group.

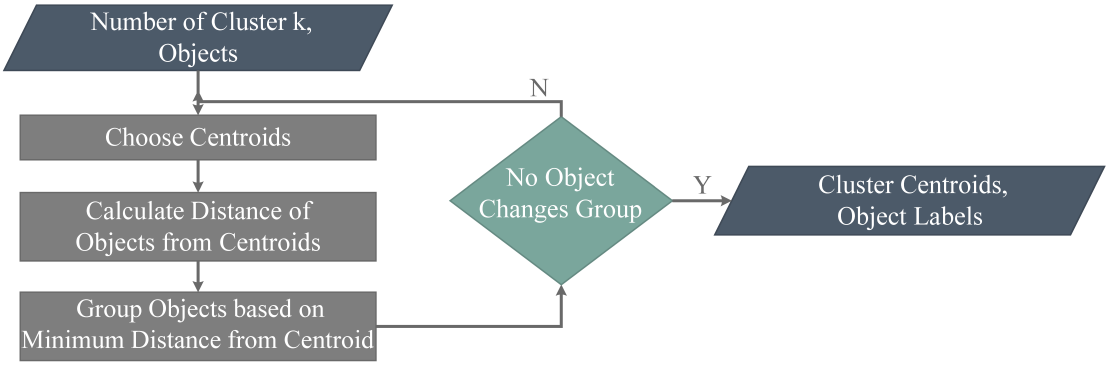

Figure 2.5: Flowchart of the K-means Clustering Algorithm.

The k-means algorithm requires a low order of memory usage and has a runtime of the order $O(n^3)$, where n is the number of objects. Moreover, k-means provides non deterministic results (different results for different runs) and requires that the number of clusters to be fixed a priori.

2.7 Summary

In this chapter, we discussed some basic concepts in probability theory, namely random variables, distribution functions, Generalized Pareto Distribution (GPD) and statistics. We then briefly discussed the notion of SVMs for statistical classification. Then, we presented how a rare failure event in analog circuits can be modelled and estimated. Finally, we presented an overview of Latin Hypercube Sampling (LHS), SPICE simulator and the k-means clustering algorithm. The intent of this chapter was to introduce the preliminaries concepts that we will be using in the remaining chapters.

Chapter 3

Classification and Estimation Methodology

In Chapter 1, we provided a general overview of the proposed methodology for modelling and estimation of analog circuit failure probability and its different processes. This chapter presents a detailed description for each process of the methodology. The methodology consists of four processes and every process is made of different stages. Figure 3.1 shows the proposed methodology processes and their respective stages in the proposed methodology. In this chapter, we explain in dedicated sections every process and all its stages as shown in the figure. We start by describing the first three processes of our methodology, i.e., *presampling*, *statistical classification* and *tail distribution modelling*. Afterward, we describe the reasons for using an iterative process of classification and tail modelling and its implementation. Finally, in the last two sections of this chapter, we present the formulation of the failure probability and discuss last process of the proposed methodology, i.e., *failure probability modelling and estimation*.

To illustrate some of the details of our methodology, we make reference to certain analog circuits, which we use as application case studies in this thesis. A detailed description of the experiments and obtained results on these case studies will be presented in Chapter 4 of this thesis.

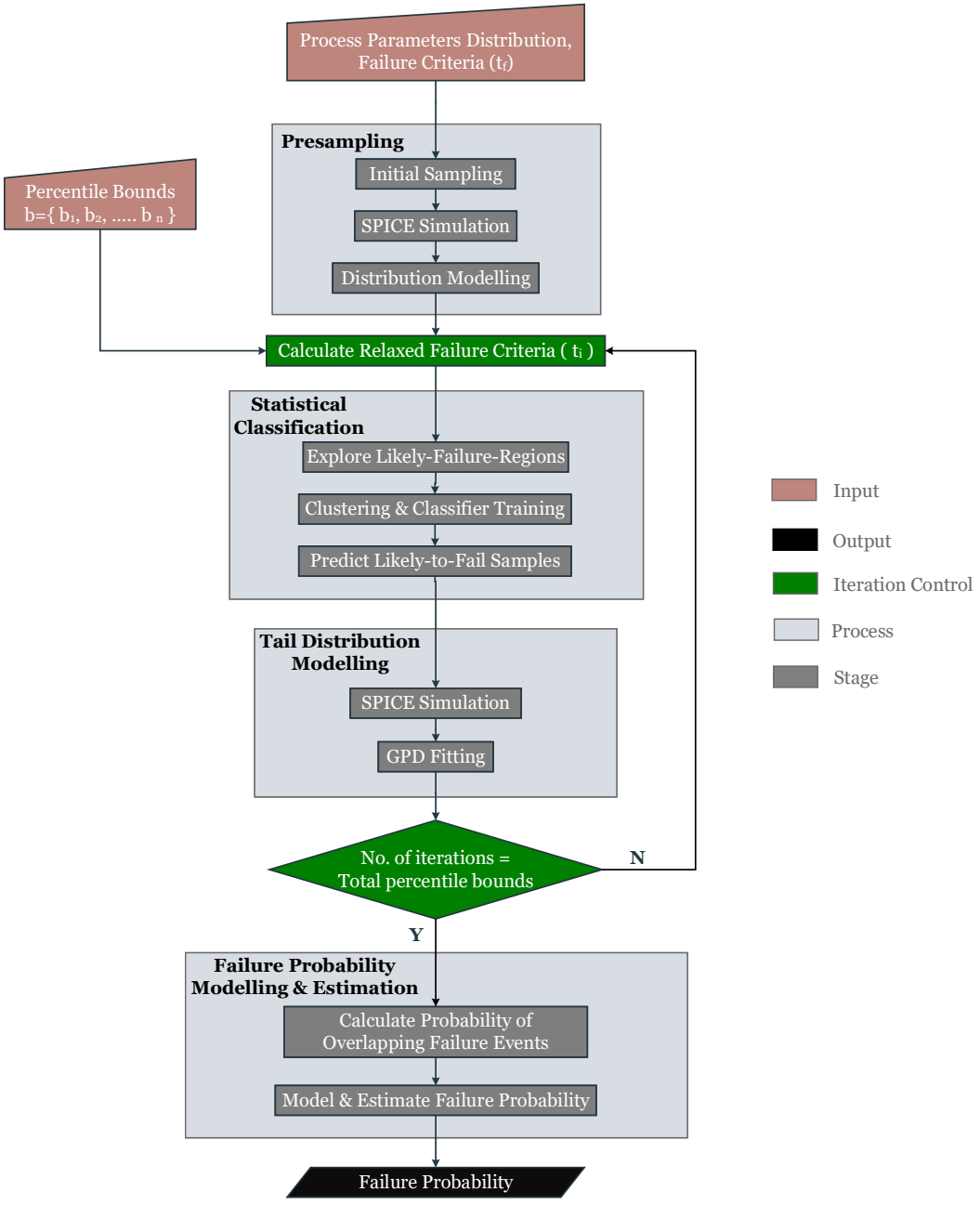

Figure 3.1: The Proposed Classification and Estimation Methodology.

3.1 Presampling

The first process of our methodology is presampling. The purpose of presampling is to sketch the approximate behavior of the analog circuit. The output from the presampling stage is later used for classifier training. Presampling itself consists of three phases:

Initial Sampling

At this point of the methodology, we are only interested in observing the circuit's approximate behavior that can be achieved using a few hundred samples of process variation parameters [43]. If the distribution of n process variation parameters $p_1, p_2, ...p_n$ is given by $S = \{P_1, P_2, ...P_n\}$. Then in order to model the PDF of the performance metrics and later for the initial classifier training, a few hundred samples, $\mathbb{S} = \{s_0, s_1, ...s_m\}$ are drawn from S. Where $s_i = \{p_{1,i}, p_{2,i}, ...p_{n,i}\}$ and m is the total number of samples drawn. Each element of $\mathbb{S}$ represents the set of values of the parameters for the circuit. This process of sampling is performed at the beginning of the methodology and is known as Initial Sampling. Samples are drawn using LHS.

Circuit Simulation

Initial Sampling provided samples $\mathbb{S}$ as output. The second phase of presampling deals with evaluating the value of performance metric Y of the circuit for the process parameters samples $\mathbb{S}$. This is achieved by performing a transistor level SPICE simulation on every sample of the set $\mathbb{S}$. SPICE simulations yield $y_0, y_1, ...y_m$ as the values of Y corresponding to the samples $s_0, s_1, ...s_m$, respectively.

Distribution Modelling

Due to process variation, Y also follows a probabilistic distribution. At this point, the PDF $f(y)$ of Y is approximated to an analytical form using the results obtained from

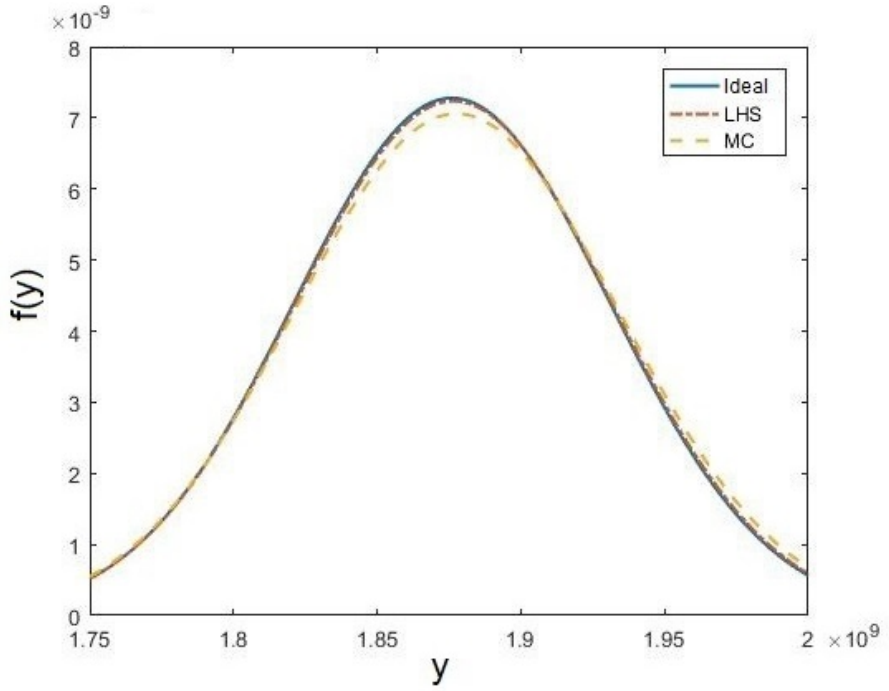

Figure 3.2: PDF Approximated using LHS and MC.

circuit simulation phase. A conventional way of fitting the PDF to an approximate analytical form is by applying moment matching methods (Section 1.2.2) on a small set of naive MC samples. In our methodology, we use samples drawn using LHS to approximate the PDF to an analytical form. Samples are more evenly spread across all possible values by the use of LHS. This helps in better PDF fitting with a smaller number of samples as compared to samples selected using the naive MC method. Figure 3.2 shows the sketches of PDF modelled using samples drawn by LHS and naive MC.

500 sample were drawn using LHS and naive MC to model the PDF of the output frequency of a ring oscillator circuit, subject to process variations of 30 parameters. Figure 3.1 clearly illustrates the advantage of using LHS for PDF fitting compared to naive MC. LHS provides a better accuracy with a smaller number of samples hence resulting in a speedup.

After $f(y)$ is determined, a relaxed failure criteria t is determined considering the

behavior of $f(y)$ and accuracy of the moment matching method used for approximating it. The value of t is such that, $F(t)$ can be accurately estimated and t lies in between the nominal value y_{nom} and the actual failure criteria t_f, in the distribution of Y, where $F(y)$ is the CDF of Y. The process of determining the value of t will be discussed later in this chapter.

3.2 Statistical Classification

In this process of our methodology, a classifier model is used to categorize a sample s_o as likely-to-fail or unlikely-to-fail. Therefore, we can skip the unlikely-to-fail samples and focus only on likely-to-fail ones. Likely-to-fail samples belong to a single or multiple regions in parameters space called Likely-to-Fail Regions (LFRs). Any sample drawn from LFR when simulated is more likely to cause circuit failure. Similarly, unlikely-to-fail samples are members of a single joint Unlikely-to-Fail Region (ULFR), which are unlikely to cause circuit failure. In this section, we first provide some background to classifiers used in other approaches and also discuss their limitations. Then, we provide details of a new classifier model developed for our methodology.

3.2.1 Background

The authors of the Statistical Blockade (SB) [42] and Recursive Statistical Blockade (RSB) [27] approaches proposed to use linear SVM (L-SVM) classifier. While in REscope [43] and Smartera [44] approaches, use of a Gaussian Radial Basis Function (GRBF) based kernel SVM (K-SVM) classifier is proposed for non-linear classification. As discussed in Section 2.2, SVMs either operate unequivocally in the input feature space giving rise to L-SVM or by using kernel mapping of feature space to higher dimensions, leading to K-SVM. L-SVM only uses dot product operation, therefore they are simple to train and use. However, they cannot be applied on non-linear data. K-SVM on other the hand can tackle the dataset which is not linearly separable. However, K-SVM is not as efficient as L-SVM. Moreover, GRBF based K-SVM

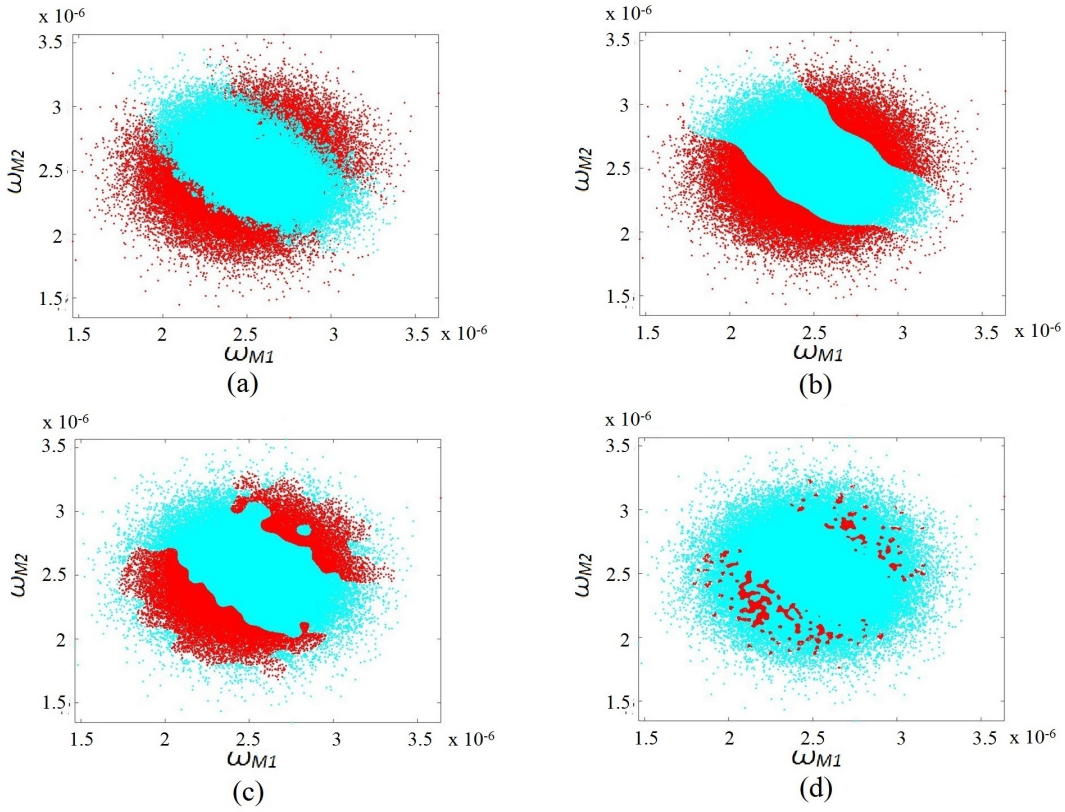

Figure 3.3: Effects of Kernel Scale γ on the GRBF based K-SVM classifier: (a) Reference Data; (b) $\gamma = 1$; (c) $\gamma = 10$; (d) $\gamma = 100$.

requires the setting of a parameter called kernel scale γ. For a lower value of γ, K-SVM is unable to capture the non-linear behavior of the boundary separating classes. While setting a higher value of γ forces K-SVM training algorithm to try harder to avoid misclassification which can consequently result in overfitting. Overfitting can significantly reduce classification accuracy. Figure 3.3 shows the effect of γ on classification accuracy of the GRBF based K-SVM classifier when used to determine LFRs of a ring oscillator circuit in the presence of two process parameters. The red region in Figure 3.3(a) represents the realistic failure region while the red region in Figures 3.3(b)-(d) represents LFR determined by the K-SVM classifier.

Based on the above discussion, it is desirable to have a classifier model which has the simplicity and efficiency of an L-SVM classifier and the high discriminative power

of a non-linear classifier such as a K-SVM classifier. The authors of [49] proposed the idea of using multiple L-SVM classifiers for non-linear classification. The authors of [49] used a mixture model of L-SVM classifiers. The approach is based on partitioning the input space (or feature space) into hyperspherical regions, in which the data is linearly separable. Experiments performed on synthetic and real world applications indicated a better training time of all L-SVM classifiers combined, with the accuracy equal to that of non-linear classifiers. This idea of using multiple L-SVM classifiers was further extended by the authors of [50, 51, 52]. All of these work employed a complex model for partitioning the feature space to make their method general in nature. These complex methods added computational cost.

In our methodology, we also propose the use of multiple L-SVM classifiers instead of a single non-linear classifier. But since our applications of statistical classification only focus on the failure probability estimation of analog circuits, assumptions can be made about the dataset. With these assumptions, complex models for partitioning a feature space are avoided and a rather simple method is adopted with significantly less computational cost. Following assumptions are drawn:

- Classification problem is binary in nature i.e. either unlikely-to-fail or likely-to-fail.
- A single ULFR exists in the feature space.
- Unlikely-to-fail samples may only be concentrated around edges of the feature space.

These assumptions were drawn by observing the behavior of different analog circuits in the presence of process variations. Based on these assumptions, we propose a classification approach in which the dataset is divided into multiple clusters. Each cluster contains both unlikely-to-fail and likely-to-fail samples which are almost linearly separable. Moreover, the unlikely-to-fail or likely-to-fail status of the training sample is evaluated based on t, the relaxed failure criteria. In remaining parts of

this section, we describe how the classifier model is developed and its working in the overall methodology.

3.2.2 Algorithm Overview

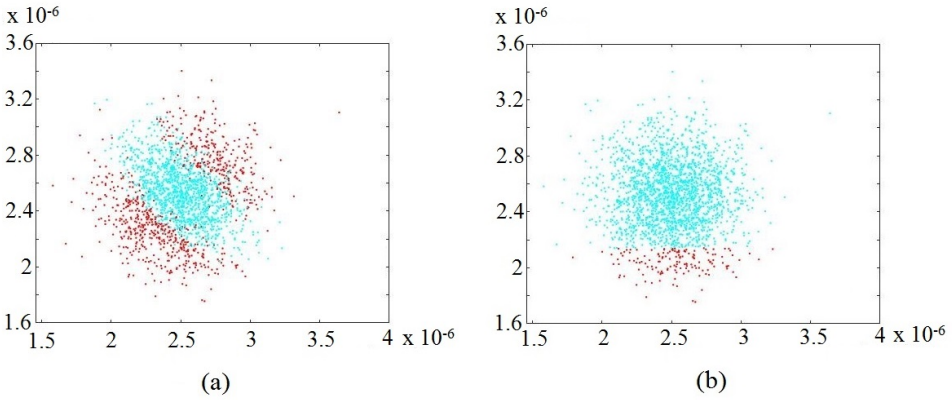

Figure 3.4: L-SVM Classification Results in the Presence of Multiple Failure Regions: (a) Reference Data; (b) Predicted Results.

A single L-SVM classifier can only linearly separate a dataset. When there exist multiple LFRs in the process parameter space, unlikely-to-fail and likely-to-fail samples cannot be separated using a single linear hyperplane. Hence, when a single L-SVM is used in this case, it fails completely. Figure 3.4 shows the classification accuracy of a single L-SVM classifier when used to categorize samples of a ring oscillator circuit in the presence of two process parameters and multiple LFRs. The red points in Figure 3.4(a) indicate samples of process parameters which generate a circuit behavior not meeting the desired specification, while the blue points represent samples generating the desired circuit behavior. It can also be seen from Figure 3.4(a) that there exist two LFRs in the parameters space. However, the red points in Figure 3.4(b) represent samples categorized likely-to-fail by the L-SVM classifier, while the blue points represent samples categorized as unlikely-to-fail. When the results presented in both Figures 3.4 (a) and (b) are compared, it can be seen that the L-SVM classifier categorized most of the failing samples as unlikely-to-fail, indicating a very

low classification accuracy.

To overcome this problem, in the first stage of our classification process, LFRs are explored. In the next stage, considering one LFR at a time, a clustering scheme is applied. In doing so, we avoid the problem of having multiple LFRs in a single cluster. Once the clusters are evaluated, multiple L-SVM classifiers are trained. Finally, in the last stage, samples are drawn and categorized as either to be unlikely-to-fail or likely-to-fail.

3.2.3 Exploring LRFs

In the first step of our classification process, LFRs are explored based on the approach proposed in [38]. LFRs represent rare events in the distribution of the performance metric Y. To effectively explore all LFRs, a large number of samples have to be drawn from the distribution of process parameters S and simulated. A surrogate model is used to overcome this problem in [38], which maps a sample s from S to the value y of the metric Y. The predicted y, obtained from the model, was then used for LFRs exploration. This approach proved useful in low dimension but failed completely in high dimensions. In our methodology, a relaxed failure criteria is chosen, hence making LFRs less rare events. Therefore, LFRs can be explored effectively with only a few hundred samples $\mathbb{S}$, simulated during the presampling stage. LFRs are explored in two stages. Figure 3.5 shows the complete procedure for exploring LFRs.

1) Basic Region Definition

A basic region represents the sub-region in the parameters space where LFRs will be explored. In Figure 3.5(a) it is the region within the two hypershperes of radius of $||r1||$ and $||r2||$ with common center s_{nom}. Where s_{nom} is the nominal value of the process parameters and $||\bullet||$ is L_2-norm function [30]. $r1$ is given by:

$$r1 = s_{r1} - s_{nom} \tag{22}$$

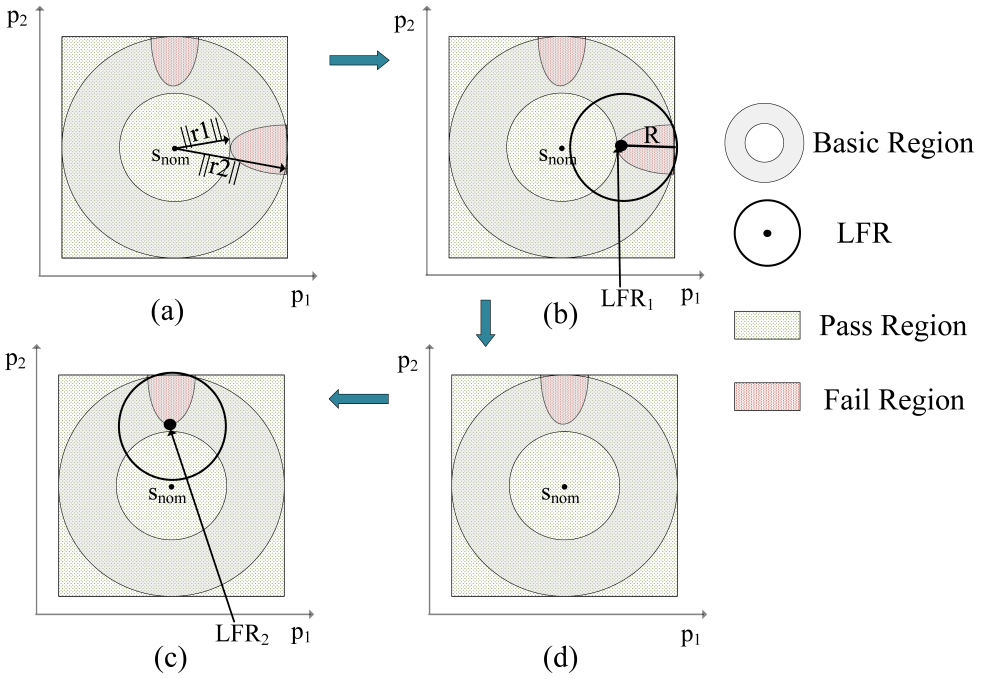

Figure 3.5: Procedure to Explore LFRs: (a) Basic Region Definition; (b) Explore First LFR; (c) Explore Second LFR; (d) Remove First.

where s_{r1} represents a sample causing circuit failure and it lies closest to s_{nom} in the space defined by S. Once $||r1||$ is evaluated, $||r2||$ is determined, and is given by:

$$r2 = s_{r2} - s_{nom} \tag{23}$$

where s_{r2} represents a sample causing circuit failure and it lies furthest away from s_{nom}.

2) LFRs Exploration

The goal of this stage is to determine the number of LFRs, x, in the basic region. To do so, only the samples residing in the basic region are selected for LFRs exploration. As shown in Figure 3.5(b), a hypercube with center s_{r1} and radius $R = ||r1|| - ||r2||$ is defined. Samples of the basic region causing circuit failure and lying within this hypercube are selected and the first LFR is determined. The selected samples are then removed from the basic region (Figure 3.5(d)). A new s_{r1} is then determined

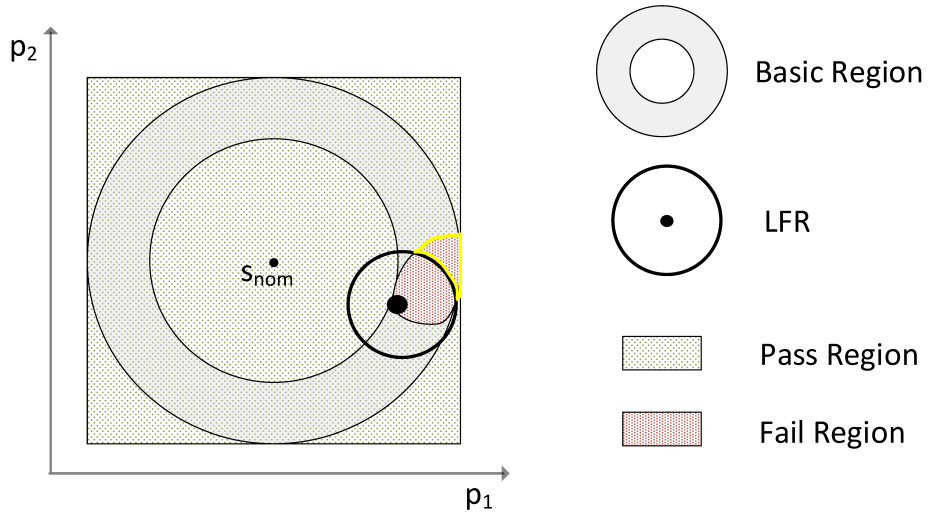

Figure 3.6: Special Condition in LFR Exploration.

from the remaining samples of the basic regions. The new value of s_{r1} is then used to define another hypercube which is shown in Figure 3.5(d). Samples that lie within the hypercube shown in Figure 3.5(c) are then selected and the second LFR is determined. This process goes on until there exist no such sample in the basic region that causes circuit failure. After that, the k-means algorithm is applied to divide all samples causing circuit failure into x groups. The output of the k-means algorithm includes x group (LFR) means and labels indicating the membership of input samples to their respective LFR.

Figure 3.6 shows a special condition which may arise during the LFR exploration. The realistic failure region is larger so that the LFR does not completely contain it. When this condition happens, the realistic failure region will be treated as several failure regions, and the other part (the yellow line in Figure 3.6) will be explored in the next iteration stage. As a result, the number of LFRs determined will be more than the number of realistic failure regions in the parameter space. The k-means algorithm is applied multiple times to overcome this problem. The k-means algorithm will provide significantly different means for different runs when the value of x is more than the number of realistic failure regions. If this situation arises, the

Algorithm 3.1 Exploring LFRs.

Require: $\mathbb{S}$, Z, s_{nom}

1: $\left[s_{r1}, ||r1||, s_{r2}, ||r2||\right] = Basic\_Region\_Paramters(\mathbb{S}, Z, s_{nom})$
2: $\left[\mathbb{S}', Z'\right] = Basic\_Region\_Sample(\mathbb{S}, ||r1||, ||r2||, s_{nom})$
3: $R = ||r1|| - ||r2||$
4: **repeat**
5: $\quad x = x + 1$
6: $\quad \left[\mathbb{S}', Z'\right] = Explore\_LFR(\mathbb{S}', Z', R, s_{r1})$
7: $\quad s_{r1} = Basic\_Region\_Paramters(\mathbb{S}', Z', s_{nom})$
8: **until** $s_{r1} \neq$ null
9: $\mathbb{S}_{Fail}$ = samples causing circuit failure
10: **repeat**
11: $\quad [S_{1,LFR}, U_1] = kmeans(\mathbb{S}_{Fail}, x)$
12: $\quad [S_{2,LFR}, U_2] = kmeans(\mathbb{S}_{Fail}, x)$
13: $\quad [S_{2,LFR}, U_3] = kmeans(\mathbb{S}_{Fail}, x)$
14: $\quad$ **if** $S_{1,LFR} \cong S_{2,LFR} \cong S_{2,LFR}$ **then**
15: $\quad\quad Stop = 1$
16: $\quad\quad S_{LFR} = S_{2,LFR}$; $U = U_1$
17: $\quad$ **else**
18: $\quad\quad x = x$ -1
19: $\quad$ **end if**
20: **until** $Stop \neq 1$
21: **return** x, S_{LFR}, $\mathbb{S}_{Fail}$, U

value of x is decreased by one and the k-means algorithm is applied multiple times using the updated value of x. This process is repeated until the means, determined by applying the k-means algorithm multiple times, are almost the same.

Implementation

A simplified implementation of the complete procedure to explore LFRs, is given in Algorithm 3.1. The inputs to the algorithm are following:

- Samples of parameters space, $\mathbb{S} = \{s_1, s_2, ..s_m\}$
- Category labels, $Z = \{z_1, z_2, ...z_m\}$. The value of z_o can either be 1 or 0 for the sample s_o. A 0 value indicates that s_o causes circuit failure while a 1 value indicates that s_o does not cause circuit failure.

- The nominal value of the process parameters, s_{nom}.

In Line 1, the algorithm outputs s_{r1}, $||r1||$, s_{r2} and $||r2||$ using the function $Basic\_Region\_Paramter()$. The inputs to the function $Basic\_Region\_Paramter()$ includes $\mathbb{S}$, Z and s_{nom}. The function $Basic\_Region\_Paramters()$ evaluates the distance of every sample in $\mathbb{S}$ from s_{nom} and determines the samples s_{r1} and s_{r2} accordingly. The function then evaluates $||r1||$ and $||r2||$. In Line 2, the algorithm determines the samples residing in the basic region using the function $Basic\_Region\_Sample()$. The input to the function $Basic\_Region\_Sample()$ includes $\mathbb{S}$, s_{nom} and the parameters of the basic region determined in Line 1. The output of the function is a group of samples $\mathbb{S}'$ and their respective category labels Z'. The distance of any sample, in the group $\mathbb{S}'$, from s_{nom} is between the interval $\big[||r1||, ||r2||\big]$. From Lines 3-8, the number of LFRs x is determined. The function $Explore\_LFR()$ in Line 6 removes those samples from the basic region that causes circuit failure and lie in the region defined by hypercube with the center s_{nom} and radius R. In Line 7, a new value of s_{r1} is determined from the remaining samples of the basic region by using the $Basic\_Region\_Paramter()$ function. Each time a new value of s_{r1} is determined, the value of x is incremented by one.

After that, the algorithm from Lines 10 to 20 iteratively determines the correct value of x by applying the k-means algorithm multiple times. Finally, the algorithm converges when a correct value of x is determined and provides the output in Line 21. The output of the algorithm includes x, LFR means $S_{LFR} = \{S_{LFR-1}, S_{LFR-2}, ...S_{LFR-x}\}$, the samples $\mathbb{S}_{Fail}$ causing circuit failure and their labels U, indicating the LFR to which they belong.

3.2.4 Clustering & Classifier Training

After the LFRs exploration stage, we move forward with the objective of clustering our feature space. We perform the clustering considering one LFR at a time. If the boundary between the LFR and ULFR is considered as a non-linear curve, dividing this curve into small segments results into portions of curves which are approximately

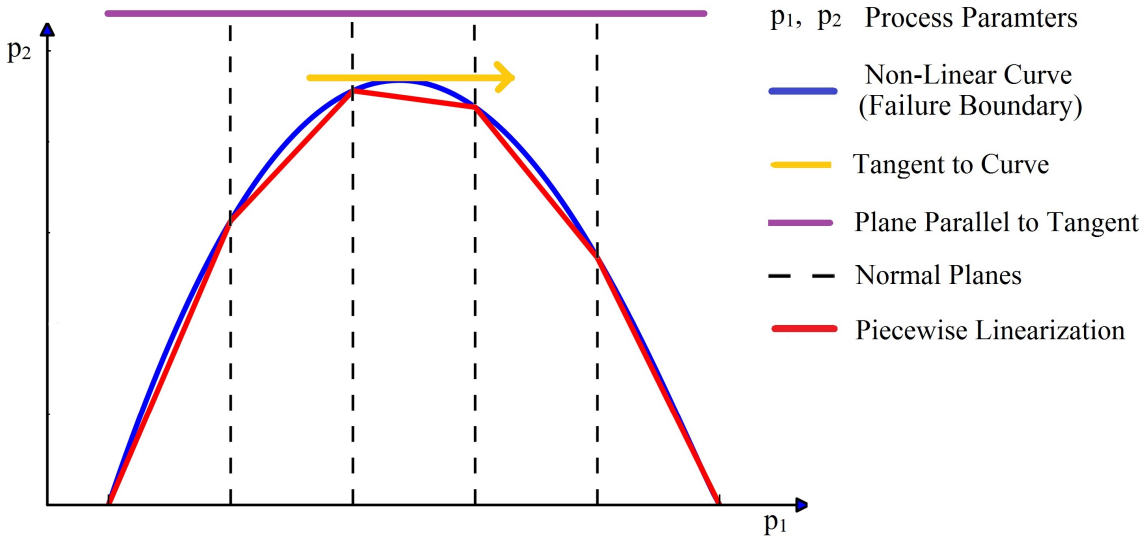

Figure 3.7: Piecewise Linearization of a Non-Linear Curve

linear. This principle is adopted in our clustering scheme. Figure 3.7 shows the principle of piecewise linearization of a non-linear curve (failure boundary) used in our classifier model. A hyperplane is determined whose direction is parallel to the curve's tangent shown in the figure. Multiple hyperplanes are then determined which are normal to the previously determined hyperplane. These hyperplanes are directed inwards the non-linear curve. If there are enough of these normal hyperplanes, a segment of the non-linear curve, lying between any two normal hyperplanes, can be considered approximately linear.

1) Cluster Centroid Evaluation

The first step, to cluster the input space defined by S, is to evaluate the centroid for each cluster. Centroids are evaluated using Algorithm 3.2. The inputs to the algorithm are following:

- Number of LFRs, x.

Algorithm 3.2 Clusters Centroids Evaluation.

Require: x, $\mathbb{S}$, s_{nom}, S_{LFR}, U

1: $[G_0, G_2, ...G_x] = Group(\mathbb{S}, U)$
2: **for** i=1 to x **do**
3: $\quad l = Calc\_Distance(s_{LFR-i}, G_i)$
4: $\quad \alpha = (s_{nom} - s_{LFR-i})/2$
5: $\quad A = \{G_0 \cup G_i\}$
6: $\quad k =$ number of samples in A
7: $\quad$ **for** j=1 to k **do**
8: $\quad\quad d_1 = ||s_{nom} - a_j||$
9: $\quad\quad d_2 = ||s_{LFR-i} - a_j||$
10: $\quad\quad d_3 = ||\alpha - a_j||$
11: $\quad\quad w_j = Find(min(D))$ ▷ where $D = \{d_1, d_2, d_3\}$
12: $\quad$ **end for**
13: $\quad J = Find(W == 3)$ ▷ where $W = \{w_1, w_2, ..w_k\}$
14: $\quad B' = Assign(A,J)$
15: $\quad J = Find(||\alpha - B'|| \leq l)$
16: $\quad B = Assign(B',J)$
17: $\quad C_i = kmeans(B, \beta)$
18: $\quad C = C \cup C_i$
19: **end for**
20: **return** C

- Samples of the parameters space, $\mathbb{S} = \{s_1, s_2, ...s_m\}$.
- Sample s_{nom}, representing nominal values of the process parameters.
- LFRs Means, $S_{LFR} = \{s_{LFR-1}, s_{LFR-2}, ...s_{LFR-x}\}$
- LFRs label, $U = \{u_1, u_2, ...u_m\}$. u can take any integer value between 0 and x. A 0 value of u indicates that a sample s belongs to ULFR. Any other value i of u indicates that a sample s belongs to i^{th} LFR with mean given by s_{LFR-i}.

In Line 1, the algorithm output $G_1, G_2, ..G_x$ that represents the group of samples having a common label defined by U. Moreover, G_0 represents the group of samples belonging to ULFR while all other groups $G_2, ..G_x$ contain samples of respective LFR. From Lines 2 to 19, the centroids $C = \{c_1, c_2, ...c_{(\beta * x)}\}$ are determined. β represents the number of clusters per LFR. The centroids are determined considering one LFR at a time. In Line 3, using the function $Calc\_Distance()$, the algorithm outputs l

which is the Euclidean distance [53] between the current LFR's mean s_{LFR-i} and the furthermost sample in the LFR. In Line 4, a midpoint α of the line joining s_{nom} and s_{LFR-i} is determined. Then in Line 5, using the samples belonging to G_0 and G_i, a group A is formed. From Lines 7 to 14, the algorithm outputs a group B' that contains samples of A which lie closest to the point α compared to the points s_{nom} and s_{LFR-i}. The function $Find()$ determines the indices of those values of a vector/matrix satisfying the condition defined in its parenthesis. The function $C = Assign(A, B)$ extracts those values of the vector A whose indices are given by the values in vector B and stores them to the vector C. From Lines 15 to 17, the k-means algorithm is applied to samples of B' whose distance from α is no more than l. The output of Line 17 includes the centroids $C_i = \{c_1, c_2, ..c_\beta\}$ for the i^{th} LFR. A superset C of the centroids is then formed by combining centroids evaluated for each LFR. It is then provided as an output by the algorithm in Line 20.

2) Cluster Assignment

After the centroids C for the clusters are determined, the samples are then assigned to the cluster associated with the nearest centroid. Algorithm 3.3 implements a simplified form of cluster assignment of the samples serving as the training data for the classifier model. The algorithm's input is the centroids C and training samples $\mathbb{S} = \{s_1, s_2, ...s_m\}$. From Lines 1 to 8, the algorithm determines, for all training samples, the distance from every centroid. The centroid having the minimum distance to any given sample is identified. Then from Lines 9 to 12, samples having the minimum distance to the centroid c_i are grouped together to form a cluster $Cluster_i$. These clusters are provided as the output of the algorithm in Line 13.

The complete process of clustering for the case of two LFRs is shown in Figure 3.8. Using Line 1 of Algorithm 3.2, the grouping of samples based on U is shown in Figure 3.8(a). The process of evaluating centroids for the first LFR using Lines 3 to 17 of Algorithm 3.2 is shown in Figures 3.8(b)-(d). Algorithm 3.2 then evaluates centroids for the second LFR and the centroids determined are shown in Figure 3.8(e). The

Algorithm 3.3 Cluster Assignment.

Require: C, $\mathbb{S}$

1: k = number of centroids in C
2: m = number of samples in $\mathbb{S}$
3: **for** $i = 1$ to m **do**
4: **for** j= 1 to k **do**
5: $d_j = ||c_j - s_i||$
6: **end for**
7: $n_i = Find(min(D))$ ▷ where $D = \{d_1, d_2, ...d_k\}$
8: **end for**
9: **for** $i = 1$ to k **do**
10: $J = Find(N == i)$ ▷ where $N = \{n_1, n_2, ...n_m\}$
11: $Cluster_i = Assign(\mathbb{S}, J)$
12: **end for**
13: **return** $Cluster_1, Cluster_2, ...Cluster_k$

dotted lines in Figure 3.8(f) represent boundaries of a cluster's region. A cluster's region is a subregion in the parameters space. Any sample that lies in the cluster's region will be assigned by Algorithm 3.3 to the cluster whose centroid is enclosed by the region. It can be seen that the failure boundary in any cluster's region is almost linear. Furthermore, the problem of getting multiple failure regions in a single cluster region is also avoided.

3) Classifier Training

Up to this stage, we have successfully evaluated the clusters. Now in the last stage, a single L-SVM classifier per cluster is trained. Cluster members given by $Cluster_i$ for the i^{th} cluster, serve as the training data for classifier. The choice of the classifier is not only limited to L-SVM classifiers, rather any linear classifier can be used. However, in our classification and estimation methodology, L-SVM classifiers are used because of their robustness in high dimensional classification problems.

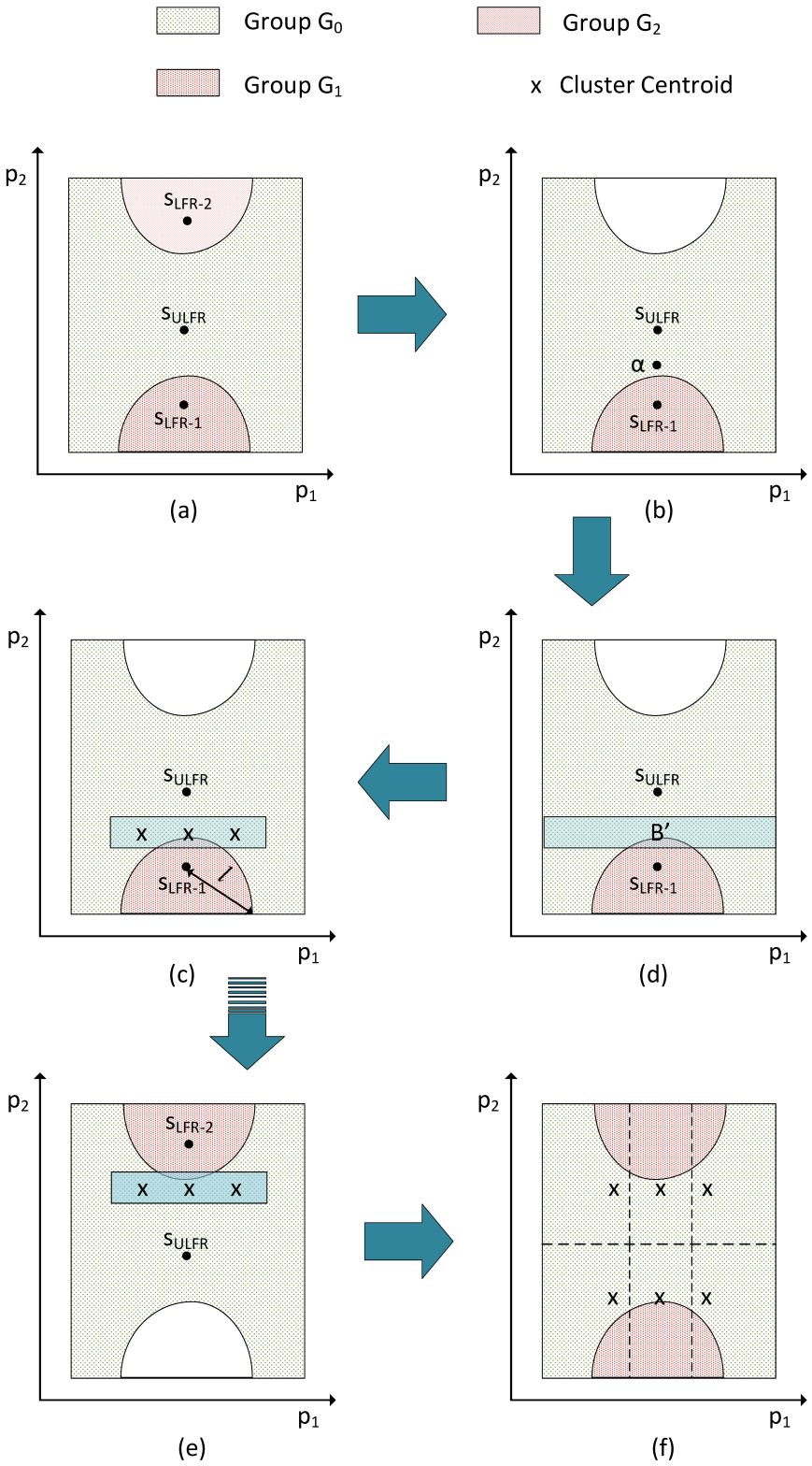

Figure 3.8: Clustering of Training Data.

3.2.5 Predicting Likely-to-Fail Samples

The last stage of the classification process is categorizing samples as unlikely-to-fail or likely-to-fail without performing SPICE simulations. Random samples $\mathbb{S}'$ are drawn from the distribution of process variations parameters S, using LHS. $\mathbb{S}'$ along with earlier evaluated clusters centroids are given as input to Algorithm 3.3. The algorithm outputs the clusters $Cluster'_1, Cluster'_2, ...Cluster'_k$. After that, samples of $Cluster'_i$ are categorized using the L-SVM classifier which was earlier trained using $Cluster_i$.

3.3 Tail Distribution Modelling

After the process of statistical classification, the next process of our methodology is to model the tail region of the performance metric's distribution. In this section, we discuss how the tail region or tail distribution of the performance metric Y is modelled by fitting the samples to GPD.

The classification process outputs two sets of samples $\mathbb{S}'_p$ and $\mathbb{S}'_f$, representing unlikely-to-fail and likely-to-fail samples, respectively. Samples belonging to $\mathbb{S}_p$ are ignored while a few hundred samples from $\mathbb{S}'_f$ are chosen for SPICE simulation, yielding values $Y_f = \{y_{f1}, y_{f2}, ...y_{fn}\}$. Samples that satisfy the condition $y_{fk} > t$, are selected for fitting the GPD. According to Equation 5, to determine the CDF $F_t(y)$ of GPD, three parameters, ϵ, μ and σ, are required. μ is the starting point of the GPD and the relaxed failure criteria t is selected as μ. Therefore, only ϵ and σ have to be estimated.

ϵ and σ can be estimated by one of the following three approaches:

- Moment Matching [41]
- Probability-weighted Moment (PWM) matching [54]
- Maximum likelihood estimation (MLE) [55]

Only the first two moments of a given sample data are used for approximating ϵ and σ by the moment matching and PWM approaches. This may lead to a mismatch

between the GPD and actual tail in higher order statistics [43]. In our methodology, the MLE approach is used, which is based on an iterative method for estimating ϵ and σ.

Maximum likelihood estimation(MLE)

The maximum likelihood estimation method estimates parameters for statistical models based on given sample data [56], such that the likelihood of selecting a training sample from the model is maximized. MLE can be considered as a two step method [57]:

- Determining a likelihood function relating the probability of given samples to parameters for the statistical models.
- Estimating those parameters values that maximize the probability of given samples in the likelihood function determined in the first step.

Using the MLE method, the parameters ϵ and σ of GPD are estimated iteratively using Newton's method, towards a maximum log likelihood function [43, 55]:

$$\log L(Y_f; \epsilon; \sigma) = -g \log(\sigma) - (1 - \epsilon) \sum_{k=1}^{g} z_k \tag{24}$$

where $z_k = -\epsilon^{-1} \log(1 - \frac{\epsilon y_{pk}}{\sigma})$

3.4 Iterative Tail Distribution Modelling

Until this section, the working of our methodology was discussed in a sequential way, without considering the iterative process. This section describes in details the iterative process for classification and tail modeling.

To understand the concept and need of the iterative method; consider an approach in which only a single relaxed failure criteria t is chosen. Choosing an optimal value for t can be a difficult task for the case of extremely rare failure events [44]. The value t is chosen such that the event $Y > t$ is not so rare and $Pb(Y > t)$ can be accurately

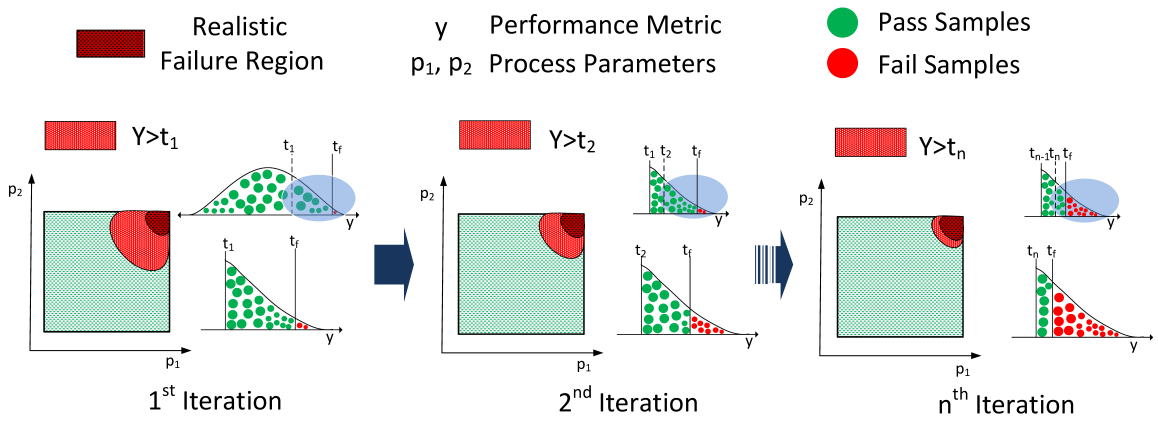

Figure 3.9: Iterative Locating of Failure Region by changing Failure Criteria.

estimated by only using a few hundred simulations to model the distribution of performance metric (Section 2.3). Let us consider an analog circuit having the failure criteria t_f, such that $Pb(Y > t_f) = 0.0001\%$. For the approach using a single relaxed failure criteria, a tail region having the probability of 1% to 10% can be chosen, if only a few hundred samples were used for modelling of the overall distribution. To model the tail, while accurately covering the event $Y > t_f$, a significant number of samples categorized as likely-to-fail, by the classifier have to be simulated. This is because of the fact that the event $Y > (t_c - t)$, given $Y > t$, is itself a rare event as $t_f - t$ is significantly large. This increases the simulation cost of the methodology. While on the other hand if only a small number of samples are simulated for tail fitting, this will result in the inaccurate modelling of extremely rare events and hence, generate an inaccurate failure probability estimation.

To overcome this problem, our methodology uses an iterative process of choosing a relaxed failure criteria, proposed by [27]. In this approach, the classification process and GPD fitting are performed iteratively using a relaxed failure criteria t_i, calculated for the i^{th} iteration. With every iteration LFRs are updated, so a new classification process is applied to capture samples of updated LFRs. These samples are then simulated to fit the GPD. For the first iteration, samples from the presampling stage are used for classifier training. While for the other iterations samples simulated for

GPD fitting in previous iterations are used. The process of iterative updating of failure region based on the new failure criteria and respective GPD fitting is shown in Figure 3.9. For the first iteration, a relaxed failure criteria t_1 is chosen which lies almost halfway between the actual failure criteria t_f and mean in the distribution of performance metric Y. By using samples residing in the region $Y > t_1$, a GPD is fitted with $\mu = t_1$. Then a new relaxed failure criteria t_2 is chosen such that $t_1 < t_2 < t_f$. A new GPD is fitted with $\mu = t_2$ using samples that lie in the region $Y > t_2$. By doing this we get a more accurate model of the region $Y > t_f$. A total of n iterations are performed. For the n^{th} iteration, we have a relaxed failure criteria t_n which lies very close to t_f and a relaxed failure region almost equal to the realistic failure region in the distribution of Y.

Calculating Relaxed Failure Criteria

The relaxed failure criteria for every iteration is calculated using percentile bounds $b = \{b_1, b_2, ...b_n\}$ by the following relation:

$$t_i = \frac{(t_f - y_{nom})b_i}{100} + y_{nom} \tag{25}$$

The value of b_i is chosen between 0% to 100%. When b_i is 0%, t_i is equal to the nominal value y_{nom} of Y while t_i is equal to t_f when b_i is chosen to be 100%. Values of b are chosen in increasing order, so with every iteration, t_i approaches more closer to t_f. While choosing values for b, the accuracy of distribution modelling approach has to be considered. For example if only 200 samples are being used to model the performance metric and the model is accurate only up to 2σ deviation from the nominal value, so the first percentile bound should be such that it generates a relaxed failure criteria which is within 2σ deviation. Similarly, other percentile bounds are chosen by considering the number of samples being used to fit GPD and accuracy of the fitting algorithm. The number of iterations performed by the methodology is equal to the number of percentile bounds. In our methodology, percentile bounds are chosen after the presampling process. An approximate behavior of the analog circuit

is available after the presampling process and analyzing it help us determine the total number of iterations required and value of percentile bound for each iteration.

3.5 Failure Probability Estimation

Once the iterative process of GDP fitting is completed, the total failure probability has to be evaluated. In Section 2.3, the process of rare event modelling using tail modelling was described. If the performance metric Y follows a Gaussian distribution with the failure criteria t_f and t_1 as the tail starting point, then the failure probability Pb_f is given by:

$$Pb_f = Pb(Y > t_1).Pb(Y > t_f|Y > t_1) = (1 - F(t_1))(1 - F_{t_1}(t_f)) \tag{26}$$

Moreover, $y_{nom} < t_1 < t_f$.
where $F_{t_1}(y)$ is the tail CDF, obtained by GPD fitting with $\mu = t_1$.
Suppose that:

$$Pb_{f1} = Pb(Y > t_f|Y > t_1) = (1 - F_{t_1}(t_f)) \tag{27}$$

Pb_{f1} represents the conditional probability of event $Y > t_f$, given by $Y > t_1$. The conditional CDF is given by $F_{t_1}(t_f)$. Now if another relaxed failure criteria t_2 is chosen, such that $t_1 < t_2 < t_f$, then using the same analogy while deriving Equation 26, we have:

$$Pb_{f1} = Pb_1(Y > t_2).Pb_1(Y > t_f|Y > t_2) = (1 - F_{t_1}(t_1))(1 - F_{t_2}(t_f)) \tag{28}$$

Substituting value of Pb_{f1} from Equation 28 in Equation 27 and then the value of $Pb(Y > t_f|Y > t_1)$ from Equation 27 in Equation 26, we get:

$$\begin{aligned} Pb_f &= Pb(Y > t_1)Pb_1(Y > t_2)Pb_1(Y > t_f|Y > t_2) \\ &= (1 - F(t_{t_1}))(1 - F_{t_1}(t_1))(1 - F_{t_2}(t_f)) \end{aligned} \tag{29}$$

Generalizing the Equation 29 for the case of the number n of failure criteria, such that $y_{nom} < t_1 < t_2 < .... < t_n$, the total failure probability becomes:

$$\begin{aligned} Pb_f &= Pb(Y > t_1) \prod_{k=1}^{n-1} \Big[Pb_k(Y > t_{k+1}) \Big] Pb_{n-1}(Y > t_f | Y > t_n) \\ &= (1 - F(t_1)) \prod_{k=1}^{n-1} \Big[(1 - F_{t_k}(t_k)) \Big] (1 - F_{t_n}(t_f)) \end{aligned} \tag{30}$$

3.6 Failure Probability Estimations for Multiple Specifications

Until now only one performance metric was considered. Most analog circuits, however, have multiple performance metrics, which have to be analyzed simultaneously. In this section, our approach for evaluating the total and individual failure probability of the analog circuits in the presence of multiple performance metrics is discussed. Every performance metric has its own specification. If a single performance metric fails to meet its specification, it is considered an overall circuit failure.

The SB [42] and RSB [27] approaches apply multiple classification processes to identify different failures resulting from each performance metric. REscope [43] and Smartera [44] use single classification processes for identifying multiple types of failure all together. However, no details were provided for the evaluation of total failure probability estimation in any of the above mentioned approaches.

In our methodology, multiple failure types are considered all together. The classification process adopted for the multiple performances is also binary, i.e., either unlikely-to-fail or likely-to-fail. Unlikely-to-fail samples are those when simulated are more likely to generate a circuit behavior that satisfies the desired specification for each of the performance metric. While for likely-to-fail samples, one or more performance metrics are likely to violate their respective specification. Once the classification process is performed, the likely-to-fail samples captured by classifier are simulated. If $Y_1, Y_2, ...Y_q$ are the performance metrics, q GPDs are fitted, one for every performance metric. If $j = 1, 2, ...q$, then, when all iterations are concluded,

we evaluate the probability $Pb_{j,f} = Pb(Y_j > t_{j,f})$ using Equation 30, for all values of j. $Pb_{j,f}$ represents the probability that a performance metric Y_j will fail to meet its desired specification, defined by the failure criteria $t_{j,f}$.

In the remaining part of this section, we derive the total failure probability of analog circuits in the presence of multiple performance metrics.

Failure Probability Formulation

Consider a case of single performance metric Y_1, its failure is an event when $y_1 > t_{1,f}$. This event is represented as the region embedded by the circle A_1 in the Venn diagram [58] representation shown in Figure 3.10(a). Hence the total failure probability of the circuit will be equal to the probability of the event represented by the circle A_1. Now for the venn diagram representation of two the performance metrics Y_1 and Y_2, we will have two circles A_1 and A_2 representing events $y_1 > t_{1,f}$ and $y_2 > t_{2,f}$, respectively, as shown in Figure 3.10(b). The events represented by the circles A_1 and A_2 are mutually exclusive [47]. Based on the criteria that a circuit fails if any of the performance metric fail, the total failure probability Pb_f is given by [47]:

$$Pb_f = Pb_{1,f} \cup Pb_{1,f} = Pb(A_1 \cup A_2) = Pb(A_1) + Pb(A_2) \tag{31}$$

where $A_g = Y_g > t_{g,f}$ with $g = 1, 2$.

Figure 3.10(c) shows the case when A_1 and A_2 may overlap, a situation which may arise in certain analog circuits [38]. For the case of not mutually exclusive events [47] as shown in Figure 3.10(c), the total failure probability is given by [47]:

$$Pb_f = Pb(A_1 \cup A_2) = Pb(A_1) + Pb(A_2) - Pb(A_1 \cap A_2) \tag{32}$$

A process parameter sample s belonging to the region represented by $A_1 \cap A_2$ will generate a circuit behavior in which both the performance metrics Y_1 and Y_2 will simultaneously violate their respective specifications. Using the inclusion-exclusion principle [59], the total failure probability for the case of q performance metrics is

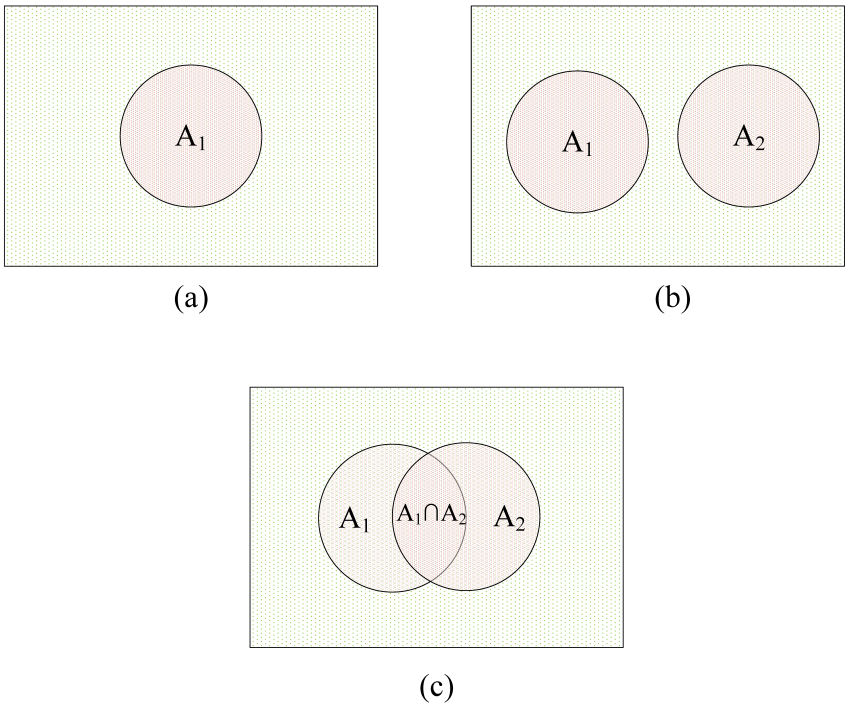

Figure 3.10: Venn Diagram Representation of Failure Event: (a) Single; (b) Double Mutually Exclusive; (c) Double Not Mutually Exclusive.

given by [59]:

$$Pb_f = Pb\left(\bigcup_{i=1}^{q} A_i\right) = \sum_{i=1}^{q} Pb(A_i) \; - \sum_{i<j} Pb(A_i \cap A_j) + \sum_{i<j<k} Pb(A_i \cap A_j \cap A_k) \; - ... + (-1)^{q-1} Pb\left(\bigcap_{i=1}^{q} A_i\right) \tag{33}$$

where $A_j = Y_j > t_{j,f}$, $j = 1, 2, ...q$.

Probability Estimation of Overlapping Failure Regions

According to Equation 33, to evaluate Pb_f accurately, apart from determining the probability of each metric failing to meet its specification, the overlapping probability also has to be determined. As mentioned at the start of this section, the metric's failure probability can be evaluated using Equation 30. Now at this stage, only the probability of overlapping region of different failure events need to be estimated. To

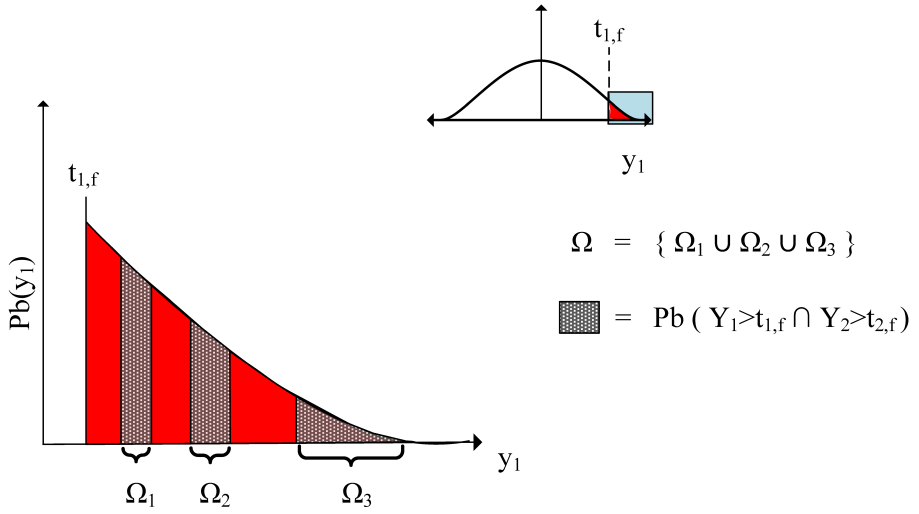

Figure 3.11: Probability of Overlapping Failure Events.

the best of our knowledge, this is the first work on classification based methods for failure probability modelling and estimation that deals with the problem of overlapping of failure events. All other work either assume that no overlapping occurs [27, 42] or ignored [43, 44] the problem of overlapping failure events. So when applied to cases with overlapping failure events, they fail to provide an accurate analysis.

To understand how the overlapping probability is calculated in our methodology, we consider the case of two metrics Y_1 and Y_2 and estimate $Pb(A_1 \cap A_2)$. Figure 3.11 shows the distribution of Y_1 in the region $y_1 > t_{1,f}$. The area shaded in grey represents $Pb(A_1 \cap A_2)$.

To estimate the regions in grey, one needs to determine Ω in the region $y_1 > t_{1,f}$. The process parameter sample s_o that generates $y_{1,o} \in \Omega$, also generates the value $y_{2,o}$ of the performance metric Y_2 such that $y_{2,o} > t_{2,f}$. By the end of the last iteration of our methodology, we have enough samples $\mathbb{S}$ of the process parameter distribution, generating $y_1 > t_{1,f}$, to effectively determine Ω. Hence no extra SPICE simulations are required. After determining Ω, using Equation 30, $Pb(A_1 \cap A_2)$ is determined. It

Algorithm 3.4 Probability Estimation of Overlapping Failure Regions.

Require: q

1: **for** $i = 2$ to q **do**
2: $Comb = NchooseK(1:q, i)$
3: $k =$ number of rows of $Comb$
4: **for** $j = 1$ to k **do**
5: $Com = ExtractRow(Comb, j)$
6: $Pb_{intersecting} = EvalOverlap(Com)$
7: $Pb_{summation} = Pb_{summation} + Pb_{intersecting}$
8: **end for**
9: $Pb_{overlap} = Pb_{overlap} + (-1)^{i-1} Pb_{summation}$
10: $Pb_{summation} = 0$
11: **end for**
12: **return** $Pb_{overlap}$

is given as:

$$Pb(A_1 \cap A_2) = Pb(Y_1 > t_{1,1}) \prod_{k=1}^{n-1} \Big[Pb_{1,k}(Y_1 > t_{1,k+1}) \Big] Pb_{1,n-1}(Y_1 = \Omega | Y_1 > t_{1,n}) \quad (34)$$

In Equation 34, only $Pb_{1,n-1}(Y_1 = \Omega | Y_1 > t_{1,n})$ needs to be estimated, as other elements have already been estimated up to this point in the methodology. $Pb_{1,n-1}(Y_1 = \Omega | Y_1 > t_{1,n})$ can be estimated using the CDF $F_{1,t_n}(y_1)$ of GDP fitted for Y_1 in the last iteration. Here Y_1 was taken as a reference metric to estimate $Pb(A_1 \cap A_2)$. Similarly, $Pb(A_1 \cap A_2)$ can also be estimated by taking Y_2 as a reference metric. In our methodology, the metric with lower index value is taken as reference metric.

To this point, we discussed how the probability of single *intersecting* event in Equation 33 is estimated. The probability of all other intersecting events is estimated in a similar manner. Algorithm 3.4 shows a simplified implementation for estimating $Pb_{overlap}$, where $Pb_{overlap}$ is given by:

$$Pb_{overlap} = -\sum_{i<j} Pb(A_i \cap A_j) + \sum_{i<j<k} Pb(A_i \cap A_j \cap A_k) - ... + (-1)^{q-1} Pb\left(\bigcap_{i=1}^{q} A_i \right) \quad (35)$$

The input to Algorithm 3.4 is the number of performance metrics q. The function $NchooseK()$ in Line 2 outputs the matrix $Comb$ containing all possible combination of q metrics taken i metrics at a time. The matrix $Comb$ has i columns and $\frac{q!}{((q-i)!i!)}$ rows.

A row $Com = \{Com_1, Com_2, ..Com_i\}$ of $Comb$ represents a possible intersecting event, where Com_j is the index of a metric. From Lines 4-8, the algorithm estimates $Pb_{summation}$, which is the value of a single summation term in Equation 35. Taking a single row of $Comb$ at a time, the probability $Pb_{intersecting}$ of an intersecting event given by the row, is estimated using the function $EvalOverlap()$ in Line 6. This function is itself an algorithm, whose working is based on estimating the probability of Ω. Figure 3.12 shows the flowchart of $EvalOverlap()$.

The input to $EvalOverlap()$ is Com, for which $Pb_{intersecting}$ is to be estimated. For simplicity, it is assumed that the function $EvalOverlap()$ has access to all samples of process parameters which were simulated and their corresponding simulation results. Apart from that, $EvalOverlap()$ also has access to the failure criteria, PDF and GPDs fitted, for all performance metrics. $EvalOverlap()$ considers the metric with the index given by Com_1 as a reference metric to estimate $Pb_{intersecting}$. Values of Y_{Com_1} are sorted in increasing order. Taking one value of Y_{Com_1} at a time, the function $EvalOverlap()$ checks whether the process parameter sample for the value is causing the metrics in Com to violate their respective specifications. If yes, the probability of the value is estimated using Equation 34 and the value of $Pb_{intersecting}$ is updated. Moreover, if more than one values satisfy this condition in a line, it is supposed that these values form a continuous range and probability of the range is estimated. When all the values of Y_{Com_1} are checked, $Pb_{intersecting}$ is provided as output.

In Line 9, $Pb_{summation}$ is added to, or subtracted from $Pb_{overlap}$ according to Equation 35. The process in Lines 1 to 9 is repeated for all summation terms in Equation 35. Finally, the algorithm provides $Pb_{overlap}$ as an output in Line 12. The failure probability Pb_f of the analog circuit is then estimated by substituting the value of $Pb_{overlap}$ in Equation 33.

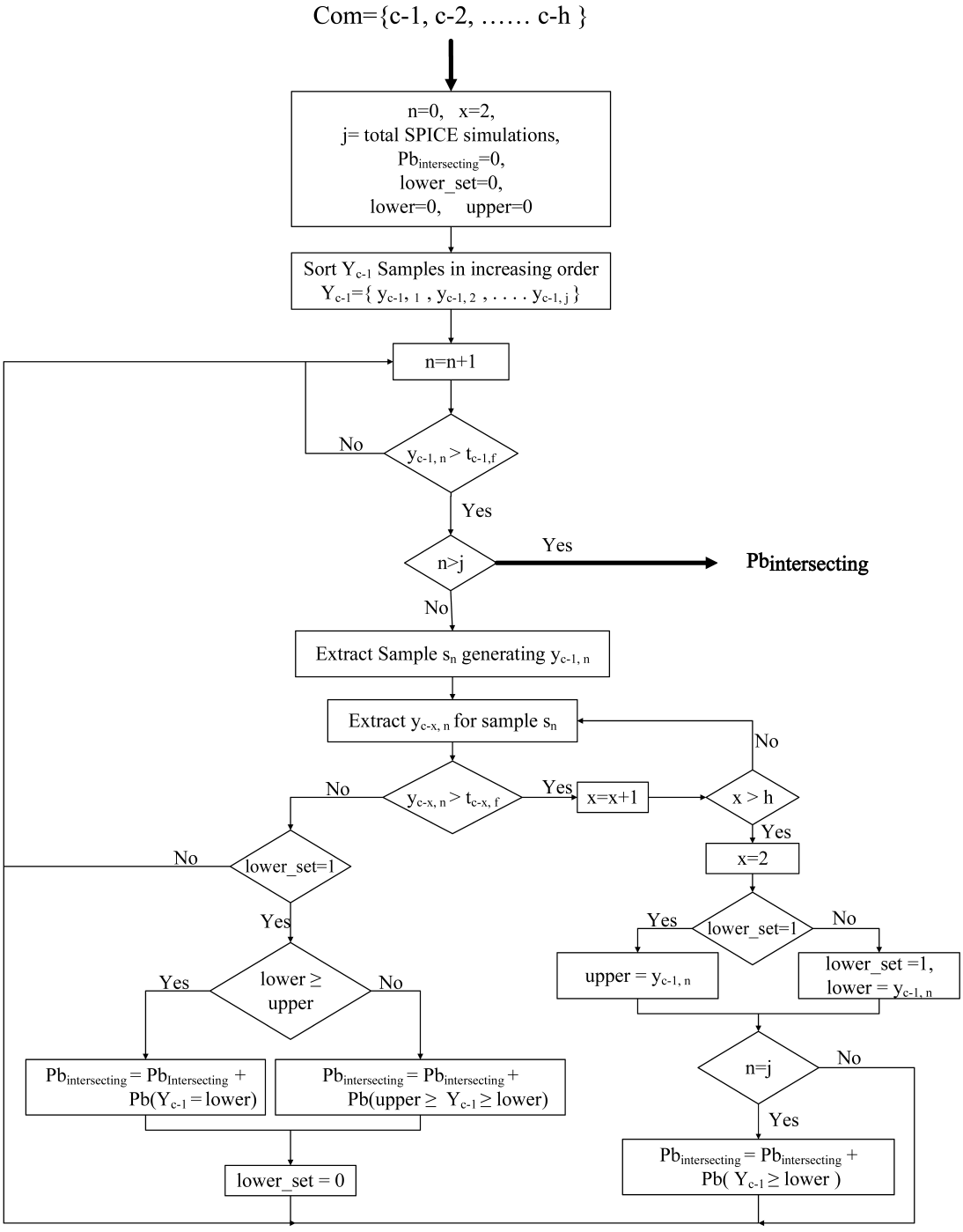

Figure 3.12: Flowchart for the *EvalOverlap*() Function.

3.7 Summary

This chapter presented a detailed description of the proposed methodology for modelling and estimating analog circuits failure probability. First, we discussed the presampling process and described its different stages. Then we discussed the statistical classification process and described a new classifier model developed for the methodology. We discussed the process of tail distribution modelling and briefly described the method for tail fitting adopted in the proposed methodology. Then we presented the reasons for using an iterative process of classification and tail modelling and briefly described its implementation. We derived the precise mathematical formula to model and estimate failure probability of analog circuits in the presence of multiple performance metrics. We also discussed the situation when different failure events might overlap and explained the algorithm developed to deal with this situation. In the next chapter, we will illustrate the application of the proposed methodology on various analog circuits to verify its effectiveness.

Chapter 4

Applications

In this chapter, we consider three real world applications to show the effectiveness of the methodology for modelling and estimating analog circuit failure probability presented in the last chapter; a 5-stage ring oscillator [60], a 3-stage opamp [61] and a 6T SRAM cell [62]. HSPICE [63] is used for the transistor level SPICE simulations. We use the commercial TSMC 65nm process [64] with BSIM4 transistor models for designing the circuits. The local mismatch variables are considered as process variables including variations in MOSFET's channel width ω, channel length L and threshold voltage V_{th} under 0V bias.

In our experiments, we use MATLAB's toolbox of statistics and machine learning [65] for sampling, distribution fitting, k-means clustering and SVM classifier training. Nominal values of performance metrics are determined by simulating the circuit in nominal condition. The naive Monte Carlo (MC) method is used as a golden reference to evaluate the accuracy and efficiency of the proposed methodology. The purpose of the proposed methodology and other advanced statistical methods is to estimate an accurate failure probability with a smaller number of SPICE simulations than the number of simulations required by the naive MC method. Therefore, in the experiments, the efficiency of the methodology is defined in terms of the speedup achieved when compared to the naive MC method. The speedup is given by the

following equation:

$$\text{Speedup } = \frac{\text{Total simulations required by the naive MC method}}{\text{Total simulations required by the proposed methodology}} \tag{36}$$

The accuracy is defined in terms of relative error of the failure probability Pb_f estimated by the methodology. The lower the relative error, the more accurate are the results. The error is given by following equation:

$$\text{Error}(\%) = \frac{|(Pb_f \text{ estimated}) - (Pb_f \text{ estimated by naive MC})|}{Pb_f \text{ estimated by naive MC}} \text{ x } 100\% \tag{37}$$

Apart from our methodology, we also implemented the approaches REScope [43] and Smartera [44] in MATLAB. Moreover, we compare our classifier model with the GRBF based K-SVM classifier. The value of kernel scale γ of the GRBF based K-SVM classifier is determined automatically by the *svmtrain* routine of MATLAB's toolbox of statistics and machine learning [65].

The number of *misclassifications* for each trained classifier model is determined. The misclassified samples represent samples of process parameters which cause circuit failure but are categorized as unlikely-to-fail by the classifier and vice-versa. The number of misclassification is used to determine the error in the classification process. The classification error is given by the following relation:

$$\text{Error}(\%) = \frac{\text{number of misclassification}}{\text{total number of samples catogrized by classifier}} \text{ x } 100\% \tag{38}$$

4.1 5-Stage Ring Oscillator

The phenomena of oscillation is found everywhere in all physical systems, especially in electronic devices. Oscillators are integral parts of all digital electronic systems that require a time reference, i.e., a clock. A perfect oscillator provides an accurate time reference. A variety of oscillators are available where the frequency band and the outputs performance in a noisy environment are different from one class of oscillators to the other. Recent designs of IC applications require oscillators with low cost and low power dissipation overshoot. The design of ring oscillators [66] using delay stages inside the IC have proved to be more useful compared to other oscillators.

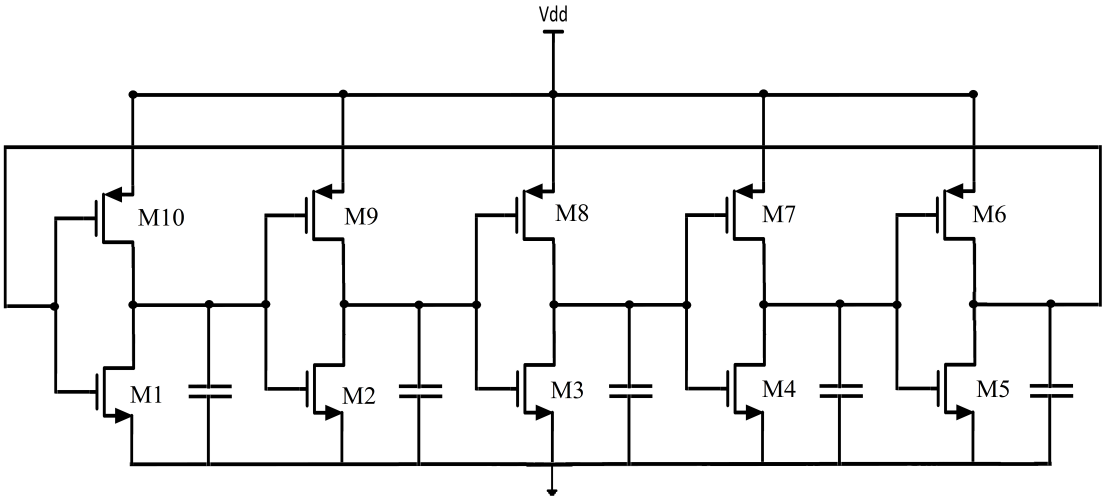

Figure 4.1: Schematics of a 5-Stage Ring Oscillator.

A ring oscillator consists of an odd number of inverters. The output of each inverter acts as input for the next one and the last output is used as the input to the first inverter. Since a single inverter evaluates the logical negation of its input, it can be proved that the last output of the chain consisting of an odd number of inverters is the logical negation of the first input. The final output arrives at a finite amount of time after the first input and the feedback of the last output to the input causes oscillation. The frequency depends on the number of stages and the time delay of the inverters. The schematic of 5-stage ring oscillator [60] circuit is shown in Figure 4.1.

We use this 5-stage ring oscillator circuit to show the validity of our methodology in high dimensional parameters space. In this experiment, the metric of interest is oscillation frequency $freq$, measured using transient analysis. The desired specification for $freq$ is between $[1.71 - 2.05]$ GHz range. The nominal frequency $freq_{nom}$ is $1.88GHz$. The process parameters consider the local variation in ω , L and V_{th} of each transistor shown in Figure 4.1, which results in a total of 30 parameters. A truncated Gaussian distribution with 3σ variation is considered for each process parameter [67].

The 5-stage ring oscillator circuit is first verified by the naive MC method, where the failure probability Pb_f converges after 600,000 simulations. Then we estimate

Pb_f using the proposed methodology. Initially, 1000 samples are drawn using LHS from the parameters space and stimulated in HSPICE. Then these simulated samples are used to model PDF of $freq$ using moment matching. $freq$ follows a gaussian distribution with mean$\approx$ $freq_{nom}$ and standard deviation $\sigma = 55.12MHz$. In this circuit, there are two failure criteria, i.e., t_f^+ and t_f^- in this circuit, representing two extremes of the desired $freq$ range. After analyzing the PDF of $freq$, the percentile bounds 30% , 60% and 90% are selected. Our methodology performs three iterations of classification and tail fitting. 700 samples predicted likely-to-fail by the classifier are simulated using HSPICE for each iteration. Two GPDs are then fitted, one for each t_f^+ and t_f^-. After that, the methodology estimates the circuit's failure probability.

Accuracy and Efficiency

The circuit failure probability Pb_f is the sum of the failure probability due to t_f^+ and t_f^-. Both criteria are for the single performance metric so no overlapping of failure events exist in this example. Hence this circuit provides one on one comparison between our methodology, naive MC, REscope and Smartera. The results are shown in Table 4.1. Our methodology is 194 times faster than the naive MC method with the relative error of only 0.2%. Moreover, our methodology provides better accuracy than REscope and Smartera thanks to the use of a better classifier model. In terms of efficiency, our methodology is better than REscope and Smartera, because we use a smarter sampling method.

Table 4.1: Failure Probability Results for the 5-Stage Ring Oscillator.

Method	**Failure Probability**	**No. of Simulation**	**Speedup**	**Error(%)**
Naive MC	1.1917e-4	600K	-	-
REscope	1.209e-4	6K	100x	1.3
Smartera	1.201e-4	3.5K	172x	0.78
Proposed Methodology	1.1951e-4	3.1K	194x	0.2

Classification Accuracy

The $freq$ specification of the 5-stage ring oscillator gives rise to multiple failure regions in the parameters space. To illustrate the capability of our proposed classifier model to handle multiple failure regions, we consider a simplified process variations model, where the source of process variations is ω of transistors M1 and M2 shown in Figure 4.1. Under this configuration, the failure regions can be clearly visualized on a 2-D space.

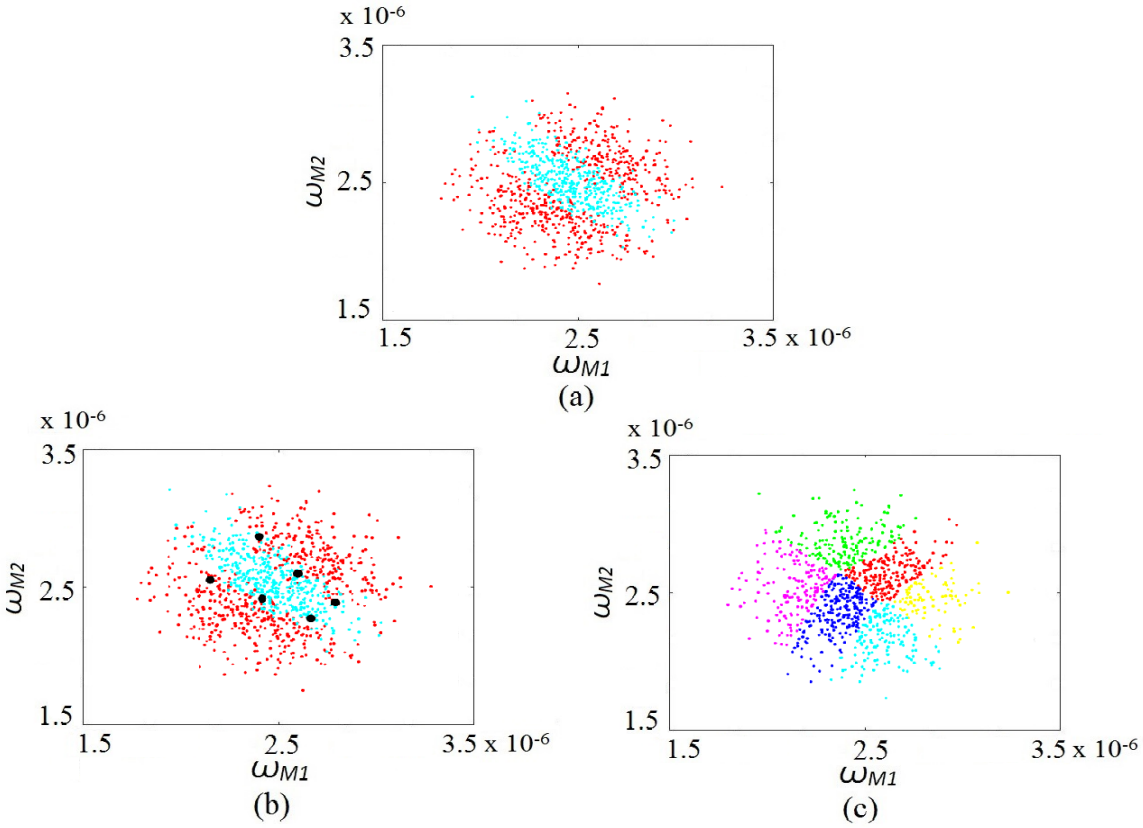

Figure 4.2: Clusters Selected by our Classifier Model: (a) Reference Data; (b) Cluster Centroid; (c) Clusters Members.

We use 1000 naive MC samples for training our classifier model. First, our classifier determines the number of failure regions and then clusters the training samples. Figure 4.2(a) shows the samples used for training our classifier model. The red points in Figures 4.2(a)-(b) are those samples that cause circuit failure while the blue points

Table 4.2: Classification Results for the Two Parameters Ring Oscillator.

Classifier	**Samples Tested**	**No. of Misclassifications**	**Error(%)**
GRBF K-SVM	80K	7937	9.9
Our Classifier Model	80K	7751	9.6

represent those samples that do not cause circuit failure. The large black dots in Figure 4.2(b) represent the cluster centroids evaluated by our classifier model using Algorithm 3.2. Points having the same colour in Figure 4.2(c) represent the samples belonging to a single cluster.

We also train the GRBF based K-SVM classifier using the same training samples. We then test the classifiers against 80,000 reference samples. The results of these experiments are presented in Table 4.2. From the table it can be seen that even in the presence of multiple failure regions, our classifier model generates more accurate results than the K-SVM classifier. The classification results of both classifiers can be visualized in Figure 4.3. The red points in Figure 4.3(a) represent those samples that cause circuit failure and vice-versa for the blue points. Similarly, the red points in Figures 4.3(b)-(c) represent those samples that are categorized likely-to-fail by the classifiers and vice-versa for the blue points.

We also performed further experiments to verify the validity of our classifier model in the high dimensional problem using samples of the 5-stage ring oscillator circuit in the presence of 30 process parameters. We used 1000 naive MC samples to train our classifier model, and used the same samples to train the GRBF based K-SVM classifier. We then tested the classifiers using 9500 reference samples. Table 4.3 summarized the results of these experiments. When the results presented in Tables 4.2 and 4.3 are compared, it can been seen that the classification error increases with increasing number of the process parameters. This increase in the error of the K-SVM classifier is more rapid than our classifier. Moreover, from Table 4.3, it can be seen that the number of misclassifications of our classifier model is quite less than that of the K-SVM classifier for the case of the 30 parameters ring oscillator.

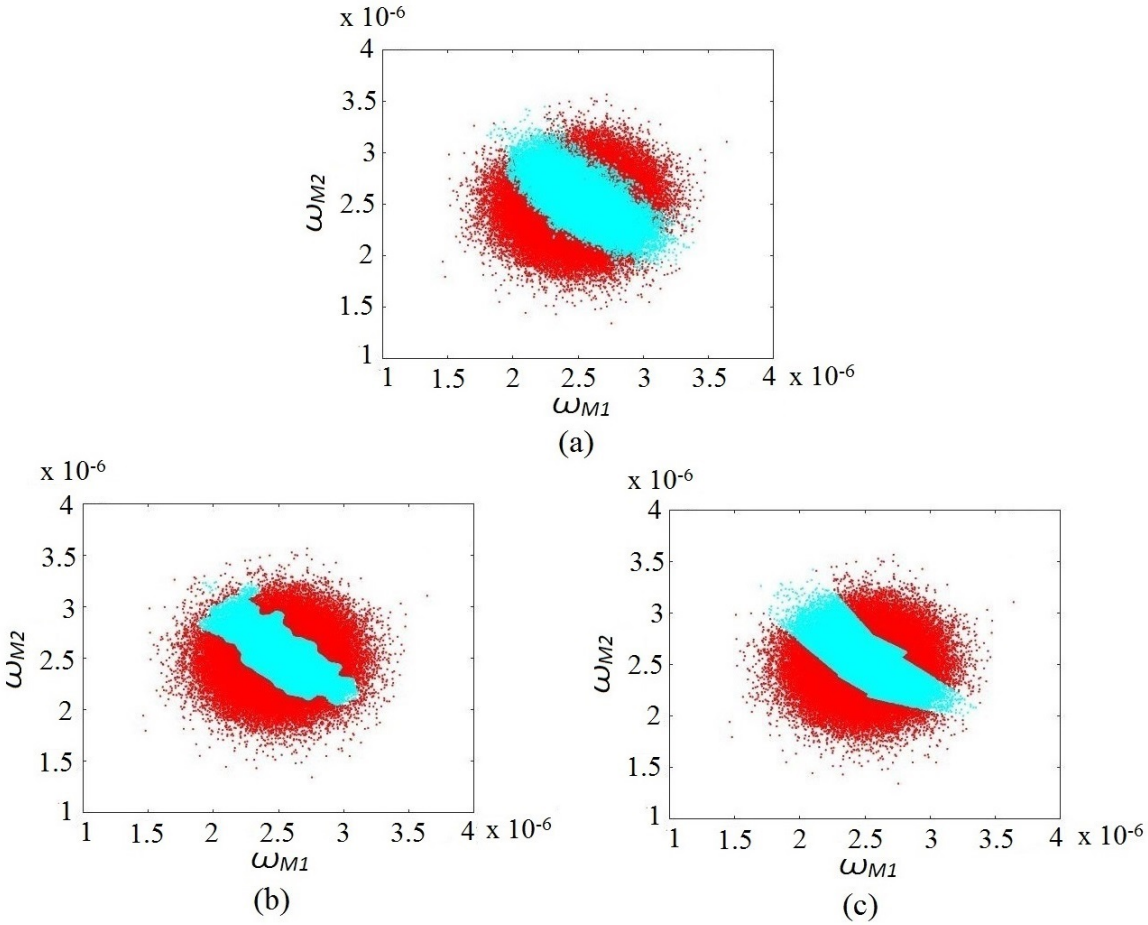

Figure 4.3: Classification Results for the Two Parameters Ring Oscillator: (a) Reference Data; (b) GRBF based K-SVM classifier; (c) Our Classifier Model.

Table 4.3: Classification Results for the 30 Parameters Ring Oscillator.

Classifier	**Samples Tested**	**No. of Misclassifications**	**Error(%)**
GRBF K-SVM	9500	1907	20.07
Our Classifier Model	9500	1605	16.89

4.2 3-Stage Opamp

Operation Amplifiers (opamp) form essential components of communication systems. They are also utilized in regulators and power management circuits and are among widely used electronics devices. The basic function of an opamp is to amplify the differential voltage between its two inputs. One of the inputs is non-inverting (+) with voltage V+ and the other is inverting (−) with voltage V−. Scaling in CMOS technology has ceaselessly challenged the established models for opamp design. Scaling in the feature size of CMOS creates faster transistors, however, the transistor's gain is reduced. In addition to these challenges, the process variations became more pronounced leading to significant offsets in opamps due to the device mismatch. In order to meet the gain specification of opamp in nano scale CMOS processes and low supply voltage, three or higher stage opamp topologies have become important. A schematic of a 3-stage opamp [61] is shown in Figure 4.4. V_p and V_m represent the opamp's non-inverting and inverting input, respectively. V_{out} represents the opamp's output. The resistors R_1 and R_2 and capacitors C_1 and C_2 are used for feedback compensation of the opamp. The capacitor C_L represents the load capacitance. V_{biasn} is the bias voltage for the first stage of the opamp.

In the sequel, we verify that our methodology is suitable for the case with multiple performances specification and overlapping failure events. The performance of the opamp is characterized by many properties, such as voltage gain (Av), gain bandwidth (GBW), etc. In our experiments, the 3-stage opamp was designed to satisfy a list of specifications shown in Table 4.4. Av is estimated by taking the ratio of the output voltage of opamp to the differential voltage at opamp's non-inverting and inverting inputs. While GBW is the frequency at which the small-signal gain equals one. Both metrics are measured during AC analysis.

To visualize the effect of overlapping failure events, we use a simplified process variations circuit model. The channel width variation in an input transistor changes the current flowing through the transistor [68], which directly affects the opamp's

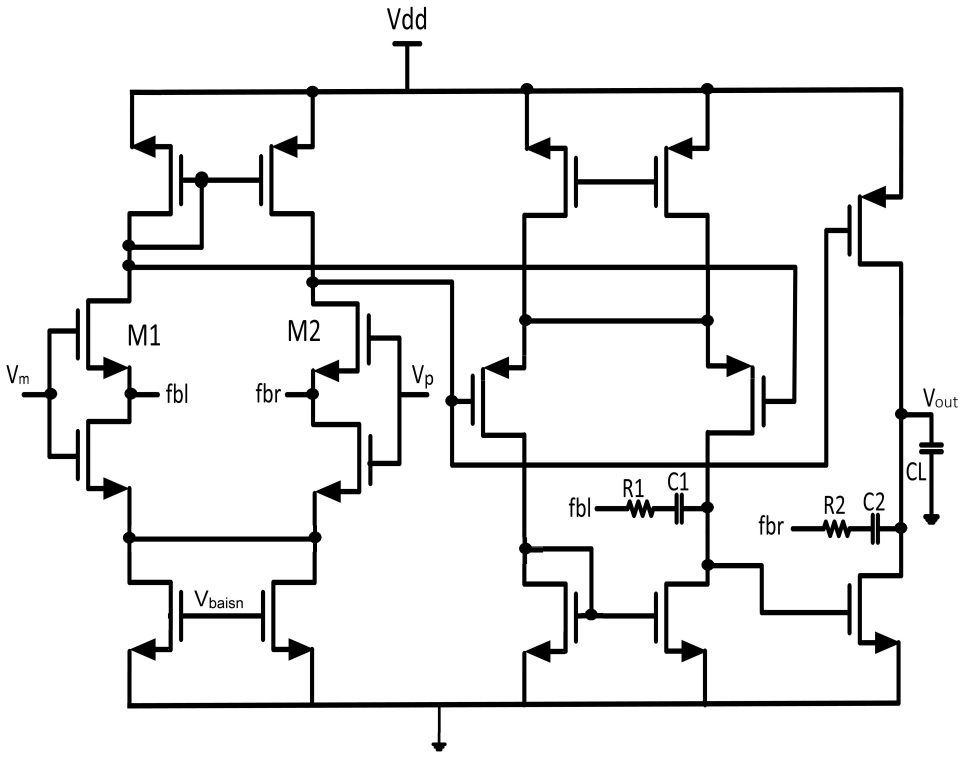

Figure 4.4: Schematics of a 3-Stage Opamp.

Table 4.4: Specifications for the 3-stage Opamp.

Performance metric	**Specification**
$Av(dB)$	≥ 37
$GBW(MHz)$	≥ 70

operation. Therefore, we select the channel width ω of the transistors M1 and M2 shown in Figure 4.4 as the process variables. We assume that both process parameters follow a truncated normal distribution having range $[\mu - 3\sigma, \mu + 3\sigma]$ [67]. Here μ represents the nominal value. The nominal values of Av and GBW are 47.7 dB and 75.7 MHz, respectively.

The circuit is first verified using the naive MC method, where the failure probability Pb_f converges after 100,000 simulations. These simulations are then used as reference data. Then, we estimate Pb_f of the circuit using our methodology. We initially draw 800 samples using LHS for the presampling process and model the PDFs

for both the Av and GBW using moment matching. The PDFs are then analyzed and the percentile bounds are selected as 30%, 60% and 90%.

The methodology estimates Pb_f in three iterations of statistical classification and GPDs fitting. At the first iteration, the methodology determines the relaxed failure criteria using the first percentile bound. Then, samples from the presampling process are used for the classifier training. After that, samples are drawn using LHS and categorized as likely-to-fail or unlikely-to-fail by the classifier. The iteration stops when 700 samples are categorized as likely-to-fail by the classifier. These samples are simulated using HSPICE. The simulation results are used to fit the first GPDs for both Av and GBW.

After that, the methodology evaluates the new relaxed failure criteria from the second percentile bound, the samples used for the GPDs fitting in the pervious iteration along with the samples from the presampling process are used to train the new classifier and the above mentioned process is repeated. Similarly, the methodology repeats the same process for the third iteration. Then the methodology estimates the failure probabilities $Pb_{Av,f}$ and $Pb_{GBW,f}$ for Av and GBW, respectively. After that, the methodology estimates the probability of overlapping failure events and then it estimates the circuit's failure probability.

Accuracy and Efficiency

Av and GBW induce failure regions which overlap in parameters space as shown in Figure 4.5. Traditionally, the failure probability of each performance metric is added to evaluate the total failure probability [38]. Since REscope and Smartera do not provide any insight on how their work deal with multiple performance metrics, the circuit's failure probability Pb_f is evaluated by adding the failure probabilities $Pb_{Av,f}$ and $Pb_{GBW,f}$, for both approaches. In Figure 4.5 it can be seen that the Av's and GBW's failure regions are almost equal. By simply adding $Pb_{Av,f}$ and $Pb_{GBW,f}$ to estimate Pb_f, results in a value of Pb_f that is almost the double to actual value. The experimental results of Pb_f, estimated for the for the 3-stage opamp circuit using the

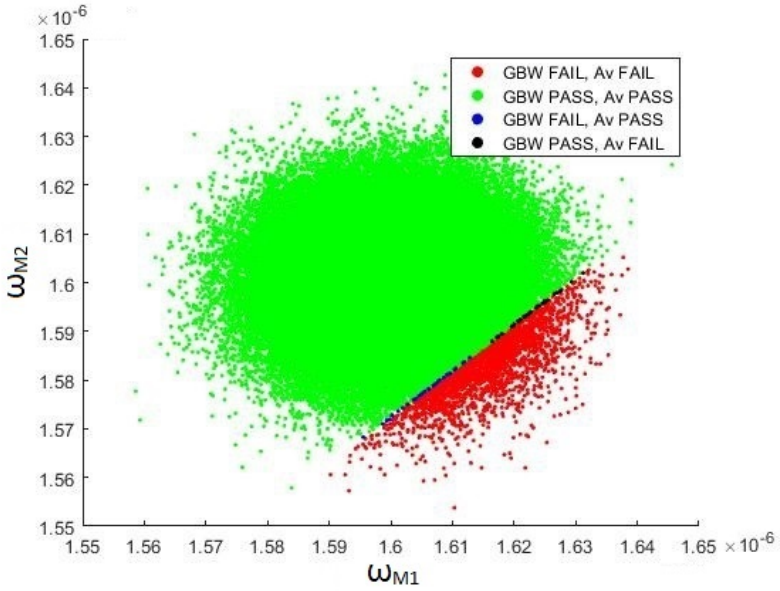

Figure 4.5: Overlapping Failure Events of the 3-stage Opamp.

Table 4.5: Failure Probability Results for the 3-Stage Opamp

Method	**Failure Probability**	**No. of Simulation**	**Speedup**	**Error(%)**
Naive MC	2.11e-2	100K	-	-
REscope	4.20e-2	3.8K	26x	98
Smartera	4.17e-2	3.5K	28x	98
Proposed Methodology	2.10e-2	3.1K	32x	0.9

naive MC method, REscope, Smartera and our methodology is shown in Table 4.5. It can be seen that our methodology is 32 times faster than the naive MC method with only 0.9% relative error. REscope and Smartera fail to estimate accurate results because both approaches are unable to deal with overlapping failure events.

Classification Accuracy

We perform experiments to verify the validity of our classifier model. We use 1000 naive MC samples to train our classifier model. The same samples are also used to train the GRBF based K-SVM classifier. The trained classifiers are then tested against 80,000 reference samples. These samples are categorized as unlikely-to-fail

Table 4.6: Classification Results for the 3-Stage opamp.

Classifier	**Samples Tested**	**No. of Misclassifications**	**Error(%)**
GRBF K-SVM	80K	59	0.074
Our Classifier Model	80K	51	0.063

or likely-to-fail and the obtained results are presented in Table 4.6. In the table, it can be seen that our classifier model is more accurate and has a lower relative error compared to the K-SVM classifier. Our classifier model generates a smaller number of misclassifications than that of the K-SVM classifier.

The classification results presented in Table 4.6 can be visualized by Figure 4.6. The red points in Figure 4.6(a) represent those samples that cause circuit failure while the blue points are the samples that do not cause circuit failure. The red points in Figures 4.6(b)-(c) represent the samples categorized as likely-to-fail by the classifier while the blue points are the samples that are categorized as unlikely-to-fail. The points in Figures 4.6(a)-(c) that lie close to the axis, represent the rare event samples in the parameters space. By comparing results shown in Figures 4.6 (a) and (b), it can be observed that most of the rare event samples, causing circuit failure, are categorized unlikely-to-fail by the K-SVM classifier. This situation may significantly affect the value of the failure probability estimated by an approach based on the K-SVM classifier.

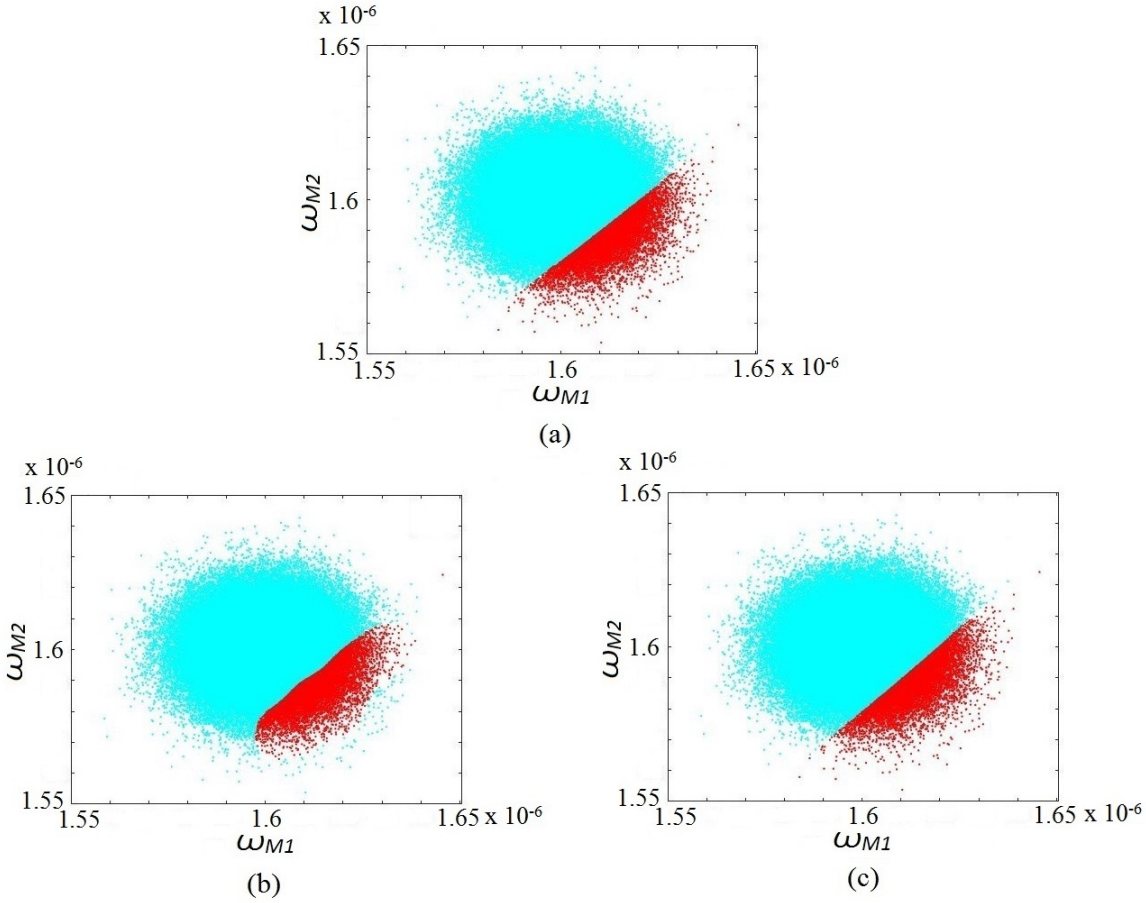

Figure 4.6: Classification Results of the 3-stage Opamp: (a) Reference Data; (b) GRBF K-SVM; (c) Our Classifier Model.

4.3 6T SRAM Cell

Static Random Access Memory (SRAM) are very popular for their large storage density and small access time. Due to the need for low power and low voltage memory design for ultrabooks, smartphones and memory cards in recent years, SRAM has become the topic of substantial research.

An SRAM cell can be constructed using a ranging number of transistors. One such implementation is using six transistors (6T). Figure 4.7 gives schematic of a 6T SRAM cell [62]. The four transistors M1, M2, M3 and M4 have two stable states, i.e., either a logic 0 or 1, and the two additional access transistors M5 and M6 serve to

control the access to the cell during read and write operations. The Word Line (WL) is used to determine whether the cell should be accessed (connected to bit line) or not, and the bit lines (BL and BL_B) are used to read/write the actual data from/to the cell. The bit lines are relatively long and have large parasitic capacitances.

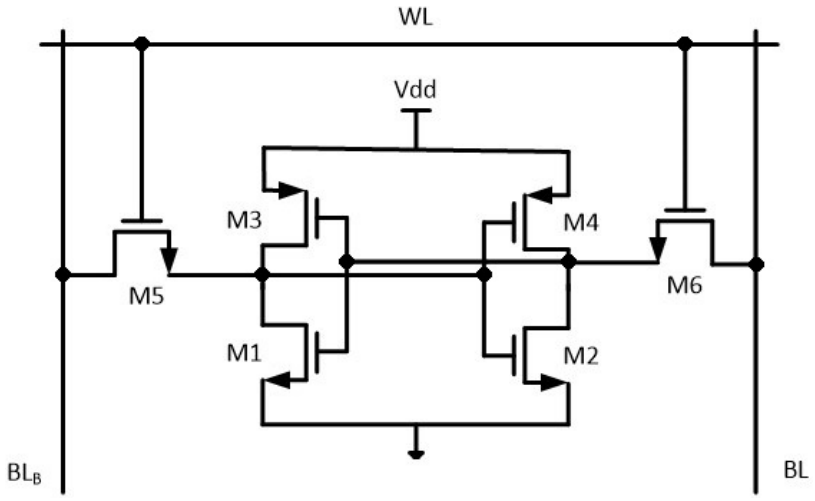

Figure 4.7: Schematic of a 6T SRAM Cell.

The increased density of SRAM in integrated devices demands the sizing to be scaled down. Due to this scaling, a lower supply voltage is required for reliable operation. This improves the power consumption but affects the performance of the SRAM, i.e, Static Noise Margin (SNM) [69] is also reduced. SNM is defined as the maximum value of DC noise voltage that can be tolerated by the SRAM cell without changing the stored bit. SNM is measured by the length of the maximum embedded square in the butterfly curves which consists of the Voltage Transfer Curves (VTC) of the two inverters in the SRAM cell. When SNM is smaller than zero, the butterfly curves collapse and data retention failure happens [69]. For a normal process variation, SNM remains positive. But extreme process variations may cause SNM to become negative, hence causing failure. It is important to evaluate extremely rare failure event in single cell to achieve high yield in SRAM chips.

We use our methodology to estimate the failure probability of 6T SRAM cell and compare the results with other methods. The performance metrics of interest

are static noise margin for read, write and hold operations. Static noise margin for read operation (RNM), is determined by setting WL to logic 1 and precharging BL and BL_B. VTC for both inverters are determined using DC analysis and RNM is estimated. For the write operation, WL and BL_B are set to logic 1 and BL is set to logic 0 when logic 0 is to be written on the SRAM cell. Again VTC for both inverters are determined using DC analysis and SNM for write operation (WNM) is estimated using these VTCs. The same process repeated for estimating the hold SNM (HNM), with WL, BL and BL_B all set to logic 0. Usually, the required specification for any noise margin has to be greater than zero. However, it can be set higher also, for a higher reliability margin [62]. The desired specification for the circuit is given in Table 4.7. The process parameters consider is the local variation in V_{th} of all transistors shown in Figure 4.7. A truncated Gaussian distribution with large variation of 6σ is considered for each process parameter [70].

Table 4.7: Specifications for the 6T SRAM Cell.

Performance metric	**Specification**
RNM	$> 0.15V$
WNM	$> 0.24V$
HNM	$> 0.26V$

Initially, the failure probability Pb_f of the circuit is estimated using the naive MC method with 50,000 samples. After that, we use our methodology to estimate Pb_f as follows. First, using LHS, 1000 samples are drawn and used for presampling. The PDF of each performance metric is modelled. After analysing the PDFs, percentile bounds, 30%, 60% and 90% are selected. After that, the analysis is performed in three iterations. The percentile bounds are used to evaluate the relaxed failure criteria for each performance metric at each iteration. Every iteration stops when the classifier selects 500 likely-to-fail samples. These samples are simulated using HSPICE to fit the three GPDs; each for the RNM, WNM and HNM. When all iterations conclude, the methodology determines individual probabilities for RNM, WNM and HNM not

meeting their desired specifications. After that, the methodology estimates the probability of overlapping failure region using Algorithm 3.4. Finally, the methodology estimates Pb_f and provides it as output.

Accuracy and Efficiency

Table 4.8: Failure Probability Results for the 6T SRAM Cell.

Method	**Failure Probability**	**No. of Simulation**	**Speedup**	**Error(%)**
Naive MC	5.122e-2	50K	-	-
REscope	5.25e-2	3K	16x	2.5
Smartera	5.24e-2	2.8K	18x	2.3
Proposed Methodology	5.133e-2	2.5K	20x	0.2

The experimental results for failure probability estimation for the 6T-SRAM cell are shown in Table 4.8. Our methodology provides results 20 times faster when compared to the naive MC method with only 0.2% relative error. Moreover, from the results presented, it can be seen that our methodology outperforms REscope and Smartera in terms of accuracy. When compared with the opamp circuit, the 6T-SRAM cell does produce an overlapping region, but this region is comparatively small. The simple addition of failure probability adopted for REscope and Smartera seem to produce reasonable results. However, the relative errors of these approaches when compared to the relative error of our methodology are still large.

Classification Accuracy

We also performed experiments to verify the validity of our classifier model for the SRAM circuit. 1000 naive MC samples are used to train our proposed classifier model and the GRBF base K-SVM classifer. These classifiers are then tested against 9500 reference samples. The results for both classifer model are shown in Table 4.9. Our classifier model has approximately 2.5% less relative error than that of the K-SVM classifier. The lower classification error allows more realistic failing samples to be

Table 4.9: Classification Results for the 6T SRAM Cell.

Classifier	**Samples Tested**	**No. of Misclassifications**	**Error(%)**
GRBF K-SVM	9500	1571	16.5
Our Classifier Model	9500	1348	14.19

captured by the classifier. As a result, a more accurate GPD fitting is achieved which results in a better estimation of the failure probability compared to the approach based on the K-SVM classifier.

4.4 Summary

In this chapter, we applied the proposed methodology for modelling and estimating failure probability on three circuits; a 5-stage ring oscillator, a 3-stage opamp and a 6T SRAM cell. The obtained results were also compared to other approaches, namely, the naive MC method, REscope and Smartera. The 3-stage opamp and 6T SRAM cell were used to verify the validity of our methodology in the presence of multiple performance metrics. The 5-stage ring oscillator was used to show that the methodology is suitable for cases in which a large number of process parameters are considered. The experimental results showed that the proposed methodology delivers many orders of speedup compared to the naive MC method with a high estimation accuracy. Furthermore, the proposed methodology was able to estimate accurate failure probabilities in the presence of multiple performance metrics while both REscope and Smartera failed completely. Moreover, in this chapter, we also verified the validity of the new classifier model developed in this thesis. We compared our classifier model with the Gaussian Radial Basis Function (GRBF) based Kernel SVM (K-SVM) classifier, by testing both classifiers on the analog circuit's process parameters dataset. The experimental results showed that our classifier has a better classification accuracy than the GRBF based K-SVM classifier.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

At nanoscale, variations in transistors parameters generate an unpredictable circuit behavior and may ultimately cause the circuit to violate constraints and fail. Thus, validating circuit reliability has become an area of great interest. Many statistical circuit simulation approaches have been proposed to evaluate the probability that a circuit does not meet the design specification. However, existing approaches lack the ability to accurately and efficiently analyze failure probability in the presence multiple performance metrics. Recognizing this, in this thesis, we proposed a methodology which can model and estimate analog circuits failure probability in the presence multiple performance metrics. The proposed methodology leverages the concepts of iterative statistical classification based approaches, to reduce the number of simulations while maintaining high estimation accuracy.

The proposed methodology consists of four processes, i.e., presampling, statistical classification, tail distribution modelling and failure probability modelling and estimation. In the first process, an approximate probabilistic behavior of the circuit is modelled by simulating a few hundred samples of process parameters. Then, in the second process, a classifier is trained to predict those samples which are likely to cause the circuit failure. A new classifier model has been developed for this process which

is based on a divide-and-conquer strategy. The classifier model uses multiple linear classifiers for the non-linear classification problem. Afterwards, in the third process, samples predicted to cause the circuit failure are simulated and the obtained results are used to model tail region of the circuit's probabilistic behavior. This tail model contains information specific to rare failure events. The second and third process are repeated iteratively to obtain a better tail model. Finally, in the last process, the circuit failure probability is modelled and estimated using the model of circuit's probabilistic behavior in tail regions. We derived the precise mathematical equation for failure probability modelling and estimation. Furthermore, the problem of overlapping of failure events in the presence of multiple performances was accurately addressed.

To show the usefulness of the proposed methodology for modelling and estimating failure probability, we applied it on three circuits; a 3-stage opamp, a 5-stage ring oscillator and a 6T SRAM cell. The obtained results were compared to the standard approach, i.e., the naive MC method. Moreover, we also compared the obtained results to other recently published statistical classification based approaches, i.e., REscope and Smartera. The experimental results showed that the proposed methodology provides many orders of speedup against the naive MC method with a high estimation accuracy. Furthermore, the proposed methodology was also able to estimate accurate failure probability for the case of overlapping failure events caused by multiple performance metrics, while REscope and Smartera failed completely. The proposed methodology was also able to estimate more accurate results using a fewer simulation runs compared to REscope and Smartera for the case of a single performance metric. Moreover, we also verified the validity of the new classifier model developed in the proposed methodology. We compared our classifier model with the Gaussian Radial Basis Function (GRBF) based Kernel SVM (K-SVM) classifier by testing both classifiers on the analog circuit dataset. The experimental results show that our classifier has a better classification accuracy than the GRBF based K-SVM classifier.

5.2 Future Work

Some of the worth mentioning future research directions based on our experience and lessons learned during the course of this thesis are outlined as follows:

1. Currently, the percentile bounds are chosen manually. An immediate extension of this thesis is to automate the process of deciding the percentile bounds and the number of iterations required for best results while maintaining high efficiency.

2. Another extension is to combine the importance sampling method and our methodology in a way that the prediction accuracy is further increased with a smaller number of training samples. By this, we expect that instead of a few thousands, only few hundred samples will be simulated in total for the analog circuit verification problem.

3. The classifier model developed for the proposed methodology can only be used for the analog circuit dataset. Another extension of this thesis is to develop a classifier model which is general in nature, i.e., the classifier can be used for any classification problem. By this, we expect that the application of the proposed methodology will extend to other fields of study, e.g., business studies, economics, etc.

4. A longer term extension of this thesis is to develop a classification process in which multiple classifiers are trained for different subsets of process parameters. Then all classifiers interact to predict the status of a sample under the influence of all process parameters. In this way the '*curse of dimensionality*' [71] for analog circuit verification will be overcome.

Bibliography

[1] Robert R. Schaller. Moore's law: past, present and future. *IEEE spectrum*, 34(6):52–59, 1997.

[2] Puneet Gupta and Fook-Luen Heng. Toward a systematic-variation aware timing methodology. In *Proceedings of Design Automation Conference*, pages 321–326. ACM, 2004.

[3] Masami Hane, T. Ikezawa, and Tatsuya Ezaki. Atomistic 3D process/device simulation considering gate line-edge roughness and poly-si random crystal orientation effects [MOSFETs]. In *Proceesings of International Electron Devices Meeting*, pages 9–5. IEEE, 2003.

[4] Ken Kundert, Henry Chang, Dan Jefferies, Gilles Lamant, Enrico Malavasi, and Fred Sendig. Design of mixed-signal systems-on-a-chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(12):1561–1571, 2000.

[5] Mark Burns, Gordon W. Roberts, and Gordon W. Roberts Mark Burns. *An Introduction to Mixed-Signal IC Test and Measurement*. Oxford University Press, 2001.

[6] Bo Liu, Georges Gielen, and Francisco Fernández. *Automated Design of Analog and High-Frequency Circuits (A Computational Intelligence Approach)*. Springer, 2014.

[7] William K. Lam. *Hardware Design Verification: Simulation and Formal Method-Based Approaches (Prentice Hall Modern Semiconductor Design Series)*. Prentice Hall PTR, 2005.

[8] Erich Barke, Darius Grabowski, Helmut Graeb, Lars Hedrich, Stefan Heinen, Ralf Popp, Sebastian Steinhorst, and Yifan Wang. Formal approaches to analog circuit verification. In *Proceedings of Design, Automation and Test in Europe*, pages 724–729. IEEE/ACM, 2009.

[9] Amith Singhee and Rob A. Rutenbar. *Novel Algorithms for Fast Statistical Analysis of Scaled Circuits*, volume 46. Springer, 2009.

[10] Farid N Najm. *Circuit Simulation*. Wiley, 2010.

[11] Carlo Jacoboni and Paolo Lugli. *The Monte Carlo Method for Semiconductor Device Simulation*. Springer, 2012.

[12] Markus Bühler, Jürgen Koehl, Jeanne Bickford, Jason Hibbeler, Ulf Schlichtmann, Ralf Sommer, Michael Pronath, and Andreas Ripp. DFM/DFY design for manufacturability and yield-influence of process variations in digital, analog and mixed-signal circuit design. In *Proceedings of Design, Automation and Test in Europe*, pages 387–392. IEEE/ACM, 2006.

[13] Wei Wu, Yi Shan, Xiaoming Chen, Yu Wang, and Huazhong Yang. FPGA accelerated parallel sparse matrix factorization for circuit simulations. In *Proceeding of International Symposium on Applied Reconfigurable Computing*, pages 302–315. Springer, 2011.

[14] Wei Wu, Fang Gong, Rahul Krishnan, Lei He, and Hao Yu. Exploiting parallelism by data dependency elimination: A case study of circuit simulation algorithms. *IEEE Journal of Design and Test*, 30(1):26–35, 2013.

[15] Xiaoming Chen, Wei Wu, Yu Wang, Hao Yu, and Huazhong Yang. An escheduler-based data dependence analysis and task scheduling for parallel circuit simulation. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 58(10):702–706, 2011.

[16] Preston G. Smith, Guy M. Merritt, and Guy M. Merritt. *Proactive Risk Management: Controlling Uncertainty in Product Development*. Productivity Press New York, 2002.

[17] Dale E. Hocevar, Michael R. Lightner, and Timothy N. Trick. A study of variance reduction techniques for estimating circuit yields. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2(3):180–192, 1983.

[18] Norman J. Elias. Acceptance sampling: An efficient, accurate method for estimating and optimizing parametric yield. *IEEE Journal of Solid-State Circuits*, 29(3):323–327, 1994.

[19] Laurence William Nagel and Donald O. Pederson. *SPICE: Simulation program with integrated circuit emphasis*. Electronics Research Laboratory, College of Engineering, University of California, Berkeley, USA, 1973.

[20] Ronald L. Iman. Latin hypercube sampling. In *Encyclopedia of Quantitative Risk Analysis and Assessment*. Wiley, 2008.

[21] Pierre L'Ecuyer. Random number generation and Quasi-Monte Carlo. In *Wiley StatsRef: Statistics Reference Online*. Wiley, 2014.

[22] Fang Gong, Hao Yu, Yiyu Shi, and Lei He. Variability-aware parametric yield estimation for analog/mixed-signal circuits: Concepts, algorithms, and challenges. *IEEE Journal of Design and Test*, 31(4):6–15, 2014.

[23] Xin Li, Jiayong Le, Padmini Gopalakrishnan, and Lawrence T. Pileggi. Asymptotic probability extraction for nonnormal performance distributions. *IEEE*

Transactions on Computer-Aided Design of Integrated Circuits and Systems, 26(1):16–37, 2007.

[24] Fang Gong, Hao Yu, and Lei He. Stochastic analog circuit behavior modeling by point estimation method. In *Proceedings of International Symposium on Physical Design*, pages 175–182. ACM, 2011.

[25] Eli Chiprout and Michel S. Nakhla. Asymptotic waveform evaluation. In *Asymptotic Waveform Evaluation*. Springer, 1994.

[26] Rahul Krishnan, Wei Wu, Fang Gong, and Lei He. Stochastic behavioral modeling of analog/mixed-signal circuits by maximizing entropy. In *Proceeding of International Symposium on Quality Electronic Design*, pages 572–579. IEEE, 2013.

[27] Amith Singhee, Jiajing Wang, Benton H. Calhoun, and Rob A. Rutenbar. Recursive statistical blockade: An enhanced technique for rare event simulation with application to SRAM circuit design. In *Proceeding of International Conference on VLSI Design*, pages 131–136. IEEE, 2008.

[28] William G. Cochran. *Sampling Techniques*. Wiley, 1977.

[29] Rouwaida Kanj, Rajiv Joshi, and Sani Nassif. Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events. In *Proceedings of Design Automation Conference*, pages 69–72. ACM, 2006.

[30] Nicolas Bourbaki. *Topological Vector Spaces*. Springer, 2013.

[31] Lara Dolecek, Masood Qazi, Devavrat Shah, and Anantha Chandrakasan. Breaking the simulation barrier: SRAM evaluation through norm minimization. In *Proceedings of International Conference on Computer-Aided Design*, pages 322–329. IEEE, 2008.

[32] Masood Qazi, Mehul Tikekar, Lara Dolecek, Devavrat Shah, and Anantha Chandrakasan. Loop flattening & spherical sampling: highly efficient model reduction techniques for SRAM yield analysis. In *Proceedings of Design, Automation and Test in Europe*, pages 801–806. IEEE/ACM, 2010.

[33] Wei Wu, Fang Gong, Gengsheng Chen, and Lei He. A fast and provably bounded failure analysis of memory circuits in high dimensions. In *Proceedings of Asia and South Pacific Design Automation Conference*, pages 424–429. IEEE, 2014.

[34] Kentaro Katayama, Shiho Hagiwara, Hiroshi Tsutsui, Hiroyuki Ochi, and Takashi Sato. Sequential importance sampling for low-probability and high-dimensional SRAM yield analysis. In *Proceedings of International Conference on Computer-Aided Design*, pages 703–708. IEEE, 2010.

[35] Andrew M. Stuart. Inverse problems: a bayesian perspective. *Acta Numerica*, 19:451–559, 2010.

[36] Reuven Y. Rubinstein and Peter W. Glynn. How to deal with the curse of dimensionality of likelihood ratios in monte carlo simulation. *Stochastic Models*, 25(4):547–568, 2009.

[37] Wei Wu, Srinivas Bodapati, and Lei He. Hyperspherical clustering and sampling for rare event analysis with multiple failure region coverage. In *Proceedings of International Symposium on Physical Design*, pages 153–160. ACM, 2016.

[38] Jian Yao, Zuochang Ye, and Yan Wang. Importance boundary sampling for SRAM yield analysis with multiple failure regions. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(3):384–396, 2014.

[39] Alexander Forrester, Andras Sobester, and Andy Keane. *Engineering Design via Surrogate Modelling: a Practical Guide*. Wiley, 2008.

[40] Donald Michie, David J. Spiegelhalter, and Charles C. Taylor. *Machine Learning, Neural and Statistical Classification*. Citeseer, 1994.

[41] Jonathan R. M. Hosking and James R. Wallis. Parameter and quantile estimation for the generalized pareto distribution. *Technometrics*, 29(3):339–349, 1987.

[42] Amith Singhee and Rob A. Rutenbar. Statistical blockade: very fast statistical simulation and modeling of rare circuit events and its application to memory design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(8):1176–1189, 2009.

[43] Wei Wu, Wenyao Xu, Rahul Krishnan, Yen-Lung Chen, and Lei He. REscope: High-dimensional statistical circuit simulation towards full failure region coverage. In *Proceedings of Design Automation Conference*, pages 1–6. ACM, 2014.

[44] Hosoon Shin, Sheldon X-D Tan, Guoyong Shi, and Esteban Tlelo-Cuautle. Rare event diagnosis by iterative failure region locating and elite learning sample selection. In *Proceeding of Latin-American Test Symposium*, pages 1–5. IEEE, 2015.

[45] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer, 2013.

[46] Shun-ichi Amari and Si Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999.

[47] Willliam Feller. *An Introduction to Probability Theory and its Applications*, volume 2. Wiley, 2008.

[48] John A. Hartigan and Manchek A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[49] Zhouyu Fu, Antonio Robles-Kelly, and Jun Zhou. Mixing linear svms for nonlinear classification. *IEEE Transactions on Neural Networks*, 21(12):1963–1975, 2010.

[50] Irene Rodriguez-Lujan, Carlos Santa Cruz, and Ramon Huerta. Hierarchical linear support vector machine. *Pattern Recognition*, 45(12):4414–4427, 2012.

[51] Quanquan Gu and Jiawei Han. Clustered support vector machines. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pages 307–315, 2013.

[52] Nikolaos Gkalelis, Vasileios Mezaris, Ioannis Kompatsiaris, and Tania Stathaki. Linear subclass support vector machines. *IEEE Signal Processing Letters*, 19(9):575–578, 2012.

[53] Michel Marie Deza and Elena Deza. Encyclopedia of distances. In *Encyclopedia of Distances*, pages 1–583. Springer, 2009.

[54] Jonathan R. M. Hosking, James R. Wallis, and Eric F Wood. Estimation of the generalized extreme-value distribution by the method of probability-weighted moments. *Technometrics*, 27(3):251–261, 1985.

[55] Jonathan R. M. Hosking. Algorithm as 215: Maximum-likelihood estimation of the parameters of the generalized extreme-value distribution. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 34(3):301–310, 1985.

[56] Johann Pfanzagl. *Parametric Statistical Sheory*. Walter de Gruyter, 1994.

[57] Gunter Löeffler and Mr Peter N. Posch. *Credit Risk Modeling using Excel and VBA*. Wiley, 2011.

[58] John Venn. *Symbolic Logic*. MacMillan, 1881.

[59] Richard A. Brualdi. *Introductory Combinatorics*. Pearson Education International, 2012.

[60] Behzad Razavi. *Design of Analog CMOS Integrated Circuits*. McGraw-Hill Education, 2000.

[61] Vishal Saxena and Russel J. Baker. Indirect compensation techniques for three-stage fully-differential op-amps. In *Proceedings of International Midwest Symposium on Circuits and Systems*, pages 588–591. IEEE, 2010.

[62] Vasudha Gupta and Mohab Anis. Statistical design of the 6T SRAM bit cell. *IEEE Transactions on Circuits and Systems I*, 57(1):93–104, 2010.

[63] Synopsys. HSPICE: Accurate Circuit Simulation. `http://www.synopsys.com/tools/Verification/AMSVerification/CircuitSimulation/HSPICE/Pages/default`, 2016.

[64] TSMC. `http://www.tsmc.com/english/dedicatedFoundry/technology/65nm.htm`, 2010.

[65] MathWorks Inc. Matlab: Statistics and Machine Learning Toolbox (R2016b). `http://www.mathworks.com/help/stats/index.html`, 2016.

[66] Jan M. Rabaey, Anantha P. Chandrakasan, and Borivoje Nikolic. *Digital integrated circuits*, volume 2. Prentice hall Englewood Cliffs, 2002.

[67] Chi-Wah Kok and Wing-Shan Tam. *CMOS Voltage References: an Analytical and Practical Perspective*. Wiley, 2012.

[68] Kerry Bernstein, Keith M Carrig, Christopher M. Durham, Patrick R. Hansen, David Hogenmiller, Edward J. Nowak, and Norman J. Rohrer. *High Speed CMOS Design Styles*. Springer, 1998.

[69] Evert Seevinck, Frans J. List, and Jan Lohstroh. Static-noise margin analysis of MOS SRAM cells. *IEEE Journal of Solid-State Circuits*, 22(5):748–754, 1987.

[70] Alberto Bosio, Luigi Dilillo, Patrick Girard, Serge Pravossoudovitch, and Arnaud Virazel. *Advanced Test Methods for SRAMs*. Springer, 2014.

[71] Claude Sammut and Geoffrey I. Webb. *Encyclopedia of Machine Learning*. Springer, 2011.